Advant® OCS
with Master software

PROFIBUS Library Editor

Version 4.7

User’s Guide

3BDS 100 585R1

Use of **DANGER**, **WARNING**, **CAUTION**, and **NOTE**

This publication includes, **DANGER**, **WARNING**, **CAUTION**, and **NOTE** information where appropriate to point out safety related or other important information.

DANGER	Hazards which could result in severe personal injury or death
WARNING	Hazards which could result in personal injury
CAUTION	Hazards which could result in equipment or property damage
NOTE	Alerts user to pertinent facts and conditions

Although **DANGER** and **WARNING** hazards are related to personal injury, and **CAUTION** hazards are associated with equipment or property damage, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, comply fully with all **DANGER**, **WARNING**, and **CAUTION** notices.

TRADEMARKS

ABB Master and **MasterPiece** are registered trademarks of ABB Asea Brown Boveri Ltd. Switzerland.

Advant, **AdvaBuild**, **AdvaCommand**, and **AdvaInform** are registered trademarks of ABB Asea Brown Boveri Ltd. Switzerland.

AdvaSoft, **Advant Station**, **AdvaBuild On-line Builder**, **Advant Controller**, and **Advant Fieldbus** are trademarks of ABB Asea Brown Boveri Ltd. Switzerland.

Microsoft and **Windows** are registered trademarks, and **Windows for Workgroups** and **Windows NT** are trademarks of Microsoft Corporation.

PROFIBUS is registered trademark of PROFIBUS Nutzerorganisation e.V. (PNO), Germany.

NOTICE

The information in this document is subject to change without notice and should not be construed as a commitment by ABB Automation Products GmbH. ABB Automation Products GmbH assumes no responsibility for any errors that may appear in this document.

In no event shall ABB Automation Products GmbH be liable for direct, indirect, special, incidental, or consequential damages of any nature or kind arising from the use of this document, nor shall ABB Automation Products GmbH be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without ABB Automation Products GmbH’s written permission, and the contents thereof must not be imparted to a third party nor be used for any unauthorized purpose.

The software described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.

Copyright © ABB Automation Products GmbH 1999

3BSE001264/E

Template: 3BSE001286/E

TABLE OF CONTENTS

Chapter 1 - Introduction

1.1 Before You Begin 1-1
1.2 Equipment Requirements 1-1
1.3 How to Use This Book 1-2
1.4 Conventions 1-3
  1.4.1 General 1-3
  1.4.2 Keys 1-3
  1.4.3 Commands and Menus 1-4
1.5 Related Documentation 1-4
1.6 Release History 1-5
1.7 Terminology 1-5
1.8 Product Overview 1-6
  1.8.1 Programming Environment 1-6
  1.8.2 General Features 1-6
  1.8.3 Support for Advant Controller 100 Series 1-7
  1.8.4 Support for Advant Controller 400 Series 1-7
1.9 User Interface 1-8

Chapter 2 - Installation

2.1 Site Planning 2-1
2.2 Installation Procedures 2-1
  2.2.1 Setup 2-1
  2.2.2 Compatibility 2-1
2.3 Shut-down and Start-up Procedures 2-1
2.4 Product Verification 2-1

Chapter 3 - Configuration/Application Building

3.1 Considerations 3-1
  3.1.1 Basic Questions 3-1
  3.1.2 System Overview 3-1
  3.1.3 Representations of PROFIBUS Configurations 3-5
3.2 Capacity and Performance 3-6
3.3 Application Start-up 3-6
3.4 Tutorial 3-6
3.5 Application Procedures 3-6

CONTENTS (continued)

Chapter 4 - Runtime Operation

4.1 Operating Overview 4-1
    4.1.1 About Preferred Mode 4-1
    4.1.2 Use of PROFIBUS Library Editor for Advant Controller 100 Series 4-2
    4.1.3 Use of PROFIBUS Library Editor for Advant Controller 400 Series 4-3
    4.1.4 Configuration and Programming in AMPL Control Configuration........ 4-4
    4.1.5 Database of Advant Controller 100 Series with PROFIBUS.......... 4-5
    4.1.6 PROFIBUS Type Circuits for Advant Controller 400 Series 4-7
4.2 Runtime Tutorial 4-10
4.3 Operating Instructions 4-10
    4.3.1 Preparation, Check of Installation and Files 4-10
    4.3.2 Editing GSD of PROFIBUS Devices and Variants 4-11
    4.3.3 Before Starting the PROFIBUS Library Editor 4-12
    4.3.4 Starting the PROFIBUS Library Editor 4-13
    4.3.5 Editing the PROFIBUS Library 4-13
    4.3.6 Compiling the PROFIBUS Library for AC 100 Series 4-14
    4.3.7 Generating Type Circuits and Template for AC 400 Series 4-15
    4.3.8 Exiting the PROFIBUS Library Editor 4-15
    4.3.9 Working with the PROFIBUS DB Element Library for AC 100 Series4-15
    4.3.10 Working with AC 400 Series Type Circuits Generated by PLE 4-16
4.4 Runtime Operation Menus 4-18
    4.4.1 Controls of PROFIBUS Library Editor 4-18
        4.4.1.1 Left Pane of Application Window 4-18
        4.4.1.2 Right Pane of Application Window 4-19
        4.4.1.3 Context Menu of Left Pane 4-40
        4.4.1.4 Context Menu of Right Pane 4-41
        4.4.1.5 Preview Window 4-41
    4.4.2 Commands of PROFIBUS Library Editor 4-44
        4.4.2.1 File Menu 4-44
        4.4.2.2 Edit Menu 4-54
        4.4.2.3 View Menu 4-57
        4.4.2.4 Actions Menu 4-58
        4.4.2.5 Help Menu 4-62

Chapter 5 - Maintenance

5.1 Error Messages 5-1

CONTENTS (continued)

Appendix A - Data Interpretation by PROFIBUS Library Editor

A.1 Mapping of GSD Keywords A-1
A.2 Rules of Creating DB Element Types A-3
A.3 Allowed Characters for DB Names A-4
  A.3.1 Allowed Characters for DB Element Callnames and Short Names A-4
  A.3.2 Allowed Characters for DB Terminal Names A-4
  A.3.3 Allowed Characters for Description of DB Element Type A-4
A.4 Standard Terminals of Slave Element Types A-5
A.5 Standard Terminals of Module Element Types A-5
A.6 Interpretation of Configuration and Diagnostic Data A-6
A.7 Interpretation Decision Tree A-9
A.8 Possible Conversions of DP Data Types A-16
A.9 Evaluation of DPV1 Data Type Bytes A-20
A.10 Evaluation of Diagnostic Data A-21
A.11 Signal Types Depending on Module Type A-21
A.12 Background of Data Type Conversion in AC 100 Series A-22

ILLUSTRATIONS

Figure 1-1. Organization of Manual 1-2
Figure 3-1. PROFIBUS Library Editor for AC 100 Series System, Overview 3-2
Figure 3-2. PROFIBUS Library Editor for AC 400 Series System, Overview 3-4
Figure 4-1. Hierarchy of DB Elements for PROFIBUS-DP Configurations 4-6
Figure 4-2. Hierarchy of Type Circuits for PROFIBUS-DP Configurations 4-8
Figure 4-3. Compact Slave, Standard Terminals Tab 4-21
Figure 4-4. Compact Slave, Process Terminals Tab 4-23
Figure 4-5. Compact Slave, UPRM Tab 4-28
Figure 4-6. Compact Slave, CFG Tab 4-30
Figure 4-7. Compact Slave, Window Names Tab 4-32
Figure 4-8. Modular Slave, Standard Terminals Tab 4-34
Figure 4-9. Modular Slave, Modules Tab 4-35
Figure 4-10. Module, Standard Terminals Tab 4-36
Figure 4-11. Module, Process Terminals Tab 4-37
Figure 4-12. Module, Signals Tab 4-38
Figure 4-13. Signal, Process Terminals Tab 4-39
Figure 4-14. Context Menu of Left Pane, Inside 4-40
Figure 4-15. Context Menu of Left Pane, Outside 4-40
Figure 4-16. Context Menu of Right Pane 4-41
Figure 4-17. Preview Window, Compact Slave 4-41
Figure 4-18. Preview Window, Modular Slave, Module Selected 4-42
Figure 4-19. Preview Window, Modular Slave, Signal Selected 4-42
Figure 4-20. Select Mode 4-44
Figure 4-21. Set the Name of the New Data Base 4-45
Figure 4-22. Open 4-46
Figure 4-23. Import 4-47
Figure 4-24. Save As 4-48
Figure 4-25. Export 4-50
Figure 4-26. Print 4-51
Figure 4-27. Print Setup 4-53
Figure 4-28. Create 4-54
Figure 4-29. Check 4-58
Figure 4-30. Generate DED for AC 100 4-59
Figure 4-31. Compile DED for AC 100 4-60
Figure 4-32. Generate Code for AC 400 4-61
Figure 4-33. PROFIBUS Library Editor: About Box 4-62
Figure A-1. Interpretation Decision Tree A-9

TABLES

Table 1-1. Related Documentation, General 1-4
Table 1-2. Related Documentation, AC 100 Series 1-4
Table 1-3. Related Documentation, AC 400 Series 1-5
Table 3-1. Checklist for PROFIBUS-DP Applications in AC 100 Series 3-6
Table 3-2. Checklist for PROFIBUS-DP Applications in AC 400 Series 3-7
Table 4-1. Compact or Modular? 4-2
Table 4-2. Data Type and Conversion (Summary) 4-25
Table 4-3. Data Group 4-25
Table 4-4. Print Range 4-52
Table A-1. Mapping of GSD Keywords A-1
Table A-2. Standard Terminals of Slave Element Types A-5
Table A-3. Standard Terminals of Slave Element Types A-6
Table A-4. GSD Byte, Numbering of Bits A-6
Table A-5. Simple Format A-7
Table A-6. Specific Format, Identifier A-7
Table A-7. Specific Format, Length Byte A-8
Table A-8. Evaluation of Input/Output in Identifier Byte (Simple Format) A-10
Table A-9. Generation of Data Type Conversion A-11
Table A-10. Evaluation of Input/Output in Identifier Byte (Specific Format) A-12
Table A-11. DP Data Type and Length “1 Byte”, Conversions A-16
Table A-12. DP Data Type and Length “1 Word”, Conversions A-17
Table A-13. DP Data Type and Length “2 Bytes”, Conversions A-17
Table A-14. DP Data Type and Length “2 Words”, Conversions A-18
Table A-15. DP Data Type and Length “n Bytes”, Conversions A-18
Table A-16. DP Data Type and Length “n Words”, Conversions A-19
Table A-17. Data Type Byte A-20
Table A-18. Evaluation of Diagnostic Data A-21
Table A-19. Signal Types Depending on Module Type A-21
Table A-20. Data on PROFIBUS-DP Interface of PM634 .... A-23
Table A-21. Data to and from the Application A-23

Chapter 1 Introduction

1.1 Before You Begin

This manual describes the PROFIBUS Library Editor version 4.7 and its usage for configuration of PROFIBUS-DP devices connected to the PROFIBUS-DP interface of both Advant Controller 100 Series and Advant Controller 400 Series.

The PROFIBUS Library Editor is attached to and delivered with the Function Chart Builder.

As compared with earlier versions, the PROFIBUS Library Editor version 4.7 contains several enhancements and new features, first of all the support of PROFIBUS-DP for Advant Controller 400 Series.

1.2 Equipment Requirements

To use the PROFIBUS Library Editor 4.7, you must have the following software and hardware:

1. Advant Engineering AMPL Control Configuration 1.7
2. Windows NT 4.0 operating system
3. You can use your own personal computer or an Advant Station 100 Series Engineering Station as a platform for running the Function Chart Builder.

NOTE

If you want to use your own PC, check whether the hardware and software requirements as stated in the Application Builder User's Guide are fulfilled. Advant Station 100 Series Engineering Stations do fulfill these requirements.

1.3 How to Use This Book

This book is organized into chapters and sections as shown in Figure 1-1.

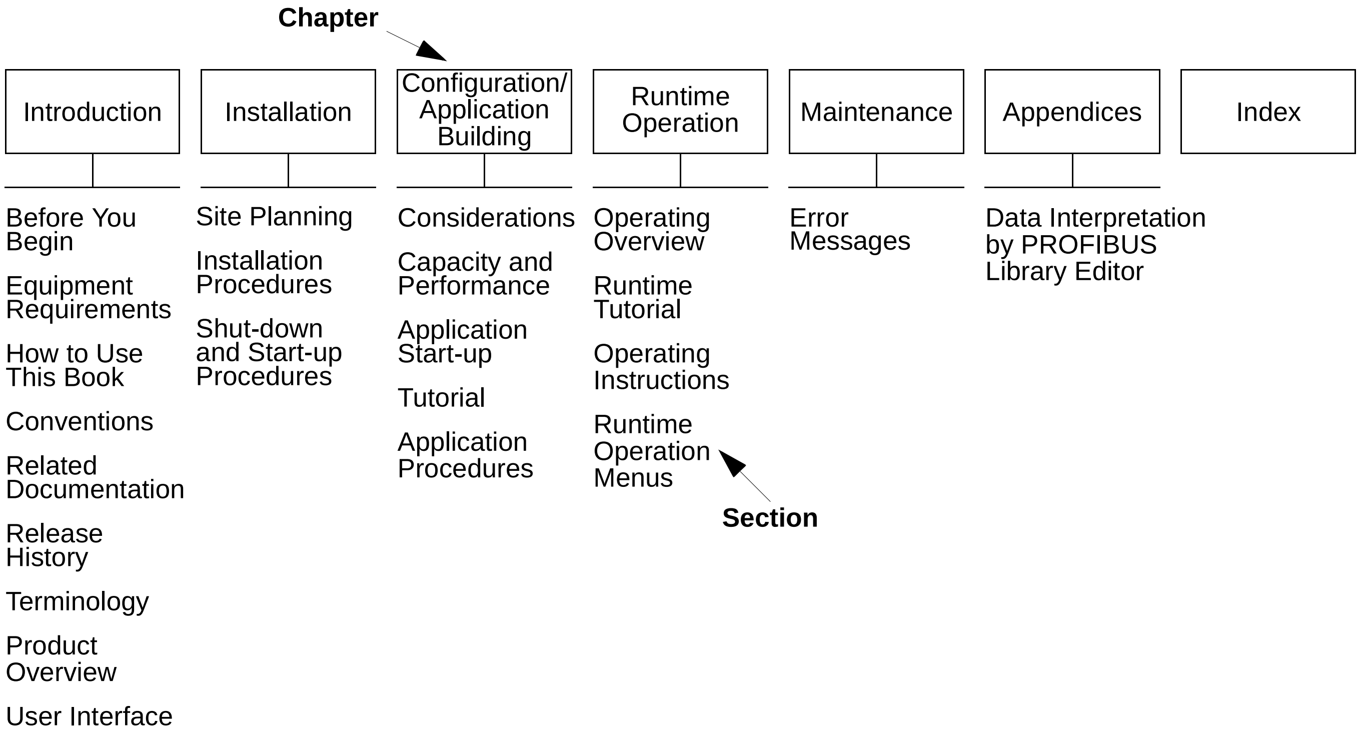


Figure 1-1. Organization of Manual

Chapter 1, Introduction

Provides you with basic information about the PROFIBUS Library Editor and this guide.

Chapter 2, Installation

Tells you how to install the PROFIBUS Library Editor.

Chapter 3, Configuration/Application Building

Gives you an overview of representation of PROFIBUS-DP in Advant Controller 100 Series and Advant Controller 400 Series systems and considerations concerning the usage of PROFIBUS Library Editor. It tells you how to start the PROFIBUS Library Editor program.

Chapter 4, Runtime Operation

Gives you an overview of product operation.

Section 4.3, Operating Instructions tells you how to perform programming operations. You will need several commands to do this. This section describes typical PROFIBUS Library Editor command sequences and explains their relationships. Look up the function you want to perform in the Table of Contents to locate the specific command sequences.

Use Section 4.4, Runtime Operation Menus as a reference to learn how to use the controls of the Title Bar, Toolbar, Status Bar, the controls in the document windows, and the menu commands of the PROFIBUS Library Editor.

Chapter 5, Maintenance

Describes the format of error messages.

Appendix A, Data Interpretation by PROFIBUS Library Editor

Contains tables with syntax description and interpretation of DB names, GSD keywords, data type, configuration data, and diagnostic data by the PROFIBUS Library Editor.

1.4 Conventions

1.4.1 General

Books, manuals, and so on, referenced in the text are written in *italic*.

Terms defined in Section 1.7, Terminology are in *italic*.

Filenames, directory names, acronyms appear in all capitals.

Words, which you have to type exactly in that form, are shown in **boldface** type, and if possible in CAPITAL letters, for example, **POWERSTN**.

Texts and system messages displayed on the screen are shown in `Courier` font, for example, `Select Options`.

Application windows, document windows, and dialog boxes are referenced by their names in **boldface** type, for example, **PC Section** window.

1.4.2 Keys

Special keyboard keys are written plain with small capitals:

- ALT, CTRL, DEL, ENTER, ESC, INS, KEYPAD 5, NUMLOCK, SCROLL LOCK, SHIFT, SPACEBAR
- BACKSPACE (for <-)
- DOWN (for down arrow, numeric keypad 2), LEFT (for left arrow, numeric keypad 4), UP (for up arrow, numeric keypad 8), RIGHT (for right arrow, numeric keypad 6)
- F1**,** F2, and so on (for function keys)
- MINUS (for '-' sign in numeric keypad), PLUS (for '+' sign in numeric keypad), ASTERISK (for '*' sign in numeric keypad)
- PGDOWN (for page down, numeric keypad 9), PGUP (for page down, numeric keypad 3)
- 1, 2,… (for keys in the upper row of the typewriter keyboard, not in the numeric keypad).

Shorthand for combined key usage, such as holding down the CTRL key and then pressing the key L:

CTRL+L.

1.4.3 Commands and Menus

Commands are written bold with capital initial:
choose **Exit** …

Menus are written bold with capital initial:
in the **File** menu …

Shorthand for commands that you choose as items from menus, for example, command **Exit** from the **File** menu, appears as **File | Exit**.

1.5 Related Documentation

The following tables contain the documentation related to PROFIBUS-DP general and in Advant Controller 100 Series and Advant Controller 100 Series:

Table 1-1. Related Documentation, General

Category	Title	Description
Software	Application Builder User's Guide	Installation of Advant Engineering components; operating instruction for APB and Advant Engineering support functions
	Function Chart Builder User's Guide	Configuring, programming, and testing of targets by FCB
Other	EN 50170	European Norm describing PROFIBUS-DP
	PROFIBUS Draft Technical Guideline PROFIBUS-DP Extensions to EN 50170 Version 1.0, December 1997	Describes PROFIBUS-DP extensions and their usage.

Table 1-2. Related Documentation, AC 100 Series

Category	Title	Description
Hardware	Advant Controller 110 PROFIBUS-DP User's Guide	Description of PROFIBUS-DP interface of Advant Controller 110
Software	AMPL Configuration AC 100 Series Reference Manual	Contains instructions for configuration, application programming, fault tracing, and maintenance.
Other	Data Base Elements Advant Controller 100 Series Reference Manual	Describes data base elements of Advant Controller 110.

Table 1-3. Related Documentation, AC 400 Series

Category	**Title**	**Description**
Hardware	Advant Controller 400 Series PROFIBUS-DP User's Guide	Description of PROFIBUS-DP interface of Advant Controller 410 and Advant Controller 450.
Software	AMPL Configuration AC 400 Series Reference Manual	Contains instructions for configuration, application programming, fault tracing, and maintenance.
Other	Data Base Elements Advant Controller 400 Series Reference Manual	Describes data base elements of Advant Controller 400 Series.
	PC Elements Advant Controller 400 Series Reference Manual	Describes PC elements of Advant Controller 400 Series.

1.6 Release History

This is the first version of the *PROFIBUS Library Editor User's Guide*, it applies to Advant Control Configuration release 1.7, consisting of 32-bit programs running under Windows NT, and describes configuration support for PROFIBUS devices in both Advant Controller 100 Series and Advant Controller 400 Series systems.

Description and usage of earlier releases of PROFIBUS Library Editor were part of earlier versions of *Advant Controller 110 PROFIBUS-DP User's Guide*. Together with release 1.7 of Advant Control Configuration, this part has been shifted out from there to build the kernel of the present user's guide.

1.7 Terminology

You should be familiar with the following terms associated with PROFIBUS-DP:

Term	**Description**
AC 100 Series	Advant Controller 100 Series
AC 400 Series	Advant Controller 400 Series
APB	Application Builder
FCB	Function Chart Builder
GSD	Device Information ("Geräte StammDaten")
ONB	On-line Builder
PLE	PROFIBUS Library Editor
PROFIBUS-DP	Process Fieldbus - Data Processing
PROFIBUS-DPV1	Process Fieldbus - Data Processing / Enhanced
TC	Type circuit

1.8 Product Overview

1.8.1 Programming Environment

AMPL Control Configuration (32-bit applications running under Windows NT) serve as **tools for configuring and programming** of Advant Open Control Systems and Advant Open Control Products. AMPL Control Configuration consists of several components, such as Application Builder, Function Chart Builder, and of options, such as AMPL PC and DB Element Libraries to be installed separately.

Generally, data base elements (DB elements) represent different hardware elements and features, such as couplers, buses, and process I/O modules. Process control elements (PC elements) represent elementary and compound process control functions.

You setup your project and manage the objects (that is, nodes, type circuit, and circuits) belonging to a certain node type in the Application Builder.

For each object, you select element libraries in form of node type and options reflecting properties and capabilities of the corresponding station, in which the application will run.

You configure and program your application with the Function Chart Builder:

- You build the data base part of your application by instantiating (that is, inserting one or several instances of) DB elements available in the selected DB element libraries or type circuits containing only DB elements.
- You build the program part by instantiating PC elements available in the selected PC element libraries or type circuits containing PC elements.
- Then, in case of Advant Controller 100 Series, you download the target code of the application to the node, start it, and test it.
- In case of Advant Controller 400 Series, you generate source code, convert it, then use the On-line Builder to download the application to the node, start it, and test it.

1.8.2 General Features

The PROFIBUS-DP interfaces of Advant Controller 100 Series and Advant Controller 400 Series allow to use PROFIBUS-DP I/O devices as distributed I/O. They conform with the PROFIBUS-DP standard EN 50170 and the enhancements defined in *PROFIBUS Technical Guideline PROFIBUS-DP Extensions to EN 50170 (DPV1) Version 2.0*. Throughout this document the term PROFIBUS-DP is used for both, except where noted.

The PROFIBUS Library Editor version 4.7 is a stand-alone tool for supporting the configuration of Advant Controller 100 Series and Advant Controller 400 Series systems containing PROFIBUS-DP interface and PROFIBUS slave stations corresponding to standards PROFIBUS-DP and DPV1.

Corresponding to the preferred representation method of PROFIBUS devices the PROFIBUS Library Editor delivers

- DB element types for Advant Controller 100 Series
- Type circuits for Advant Controller 400 Series.

The PROFIBUS Library Editor is a 32-bit application running under Windows NT. It works on Microsoft Access 97 data base files (*.MDB files).

The PROFIBUS Library Editor is part of AMPL Control Configuration. It consists of an executable and a model data base. It is attached to and delivered with the Function Chart Builder and uses the description compiler also delivered with the Function Chart Builder.

The PROFIBUS Library Editor allows to extract information from *.GSD files at creation of DB element types and type circuits.

The PROFIBUS Library Editor offers export functions to build a stock of PROFIBUS device type descriptions (PROFIBUS element library) and import functions to facilitate their reuse.

1.8.3 Support for Advant Controller 100 Series

The PROFIBUS Library Editor version 4.7 supports generation and installation of DB element types (DB element descriptions) concerning slave stations corresponding to standards PROFIBUS-DP and DPV1.

When working with AMPL Control Configuration for Advant Controller 100 Series, PROFIBUS master and slave stations are represented by DB elements (DB element instances). With the PROFIBUS Library Editor you can create DB element types for the description of compact PROFIBUS slave stations, modular PROFIBUS slave stations, and modules and signals of the latter, corresponding to your set of PROFIBUS devices.

The DB element types are collected and compiled to appropriate DB element library, which can be used by the Function Chart Builder when configuring and programming your application for a given node type.

1.8.4 Support for Advant Controller 400 Series

The PROFIBUS Library Editor version 4.7 supports generation of Advant Controller 400 Series type circuits for use with the Function Chart Builder and AMPL source code templates for work with the On-line Builder.

The PROFIBUS Library Editor generates a DB type circuit and a PC type circuit for every PROFIBUS device type.

The DB type circuit contains a PBSD element. It carries the name of the device type and the configuration data. This type circuit has to be instantiated once by the Function Chart Builder when configuring and programming your application.

The PC type circuit contains a PBS element and several PB-R, PB-S, and PB-DIAG elements. The former assigns the device type to a unique bus and station number pair. The latter provide the input, output, and diagnostic PC terminals to be connected in the application. This type circuit has to be instantiated by the Function Chart Builder as many times as many devices of this type are connected to the given AC 400 Series controller.

Furthermore, the PROFIBUS Library Editor generates AMPL source code (*.AAX and *.BAX files) that can be used as template for editing an application with the On-line Builder.

1.9 User Interface

The PROFIBUS Library Editor is a Windows application program with typical graphic user interface and look-and-feel of Windows programs. It presents the different objects in windows, offers menus and dialog boxes. You choose the commands by accelerator keys and shortcuts on the keyboard or by mouse actions.

See detailed information on the graphic user interface and on interacting with Windows programs in the user´s guide of Windows installed on your engineering station, for example,

Basics and Installation Microsoft Windows NT Workstation Version 4.0, Microsoft Corporation, Document No. 94945-0996.

Or select **Help** from the **Start** menu of Windows NT and then look for relevant topics in Windows NT Help.

Chapter 2 Installation

2.1 Site Planning

Product Guide - Advant Engineering Products for environmental considerations,
Advant Station 100 Series Engineering Station User's Guide.
If you are working on a PC of your own, consult its user documentation.

2.2 Installation Procedures

2.2.1 Setup

The PROFIBUS Library Editor is attached to Function Chart Builder and is installed automatically together with the latter.

As hardware and software platform for running the PROFIBUS Library Editor, you can use an Advant Station 100 Series Engineering Station with installed Windows NT operating system.
Or you can use an appropriate PC of yours with installed Windows NT.
The installation is described in the *Function Chart Builder Release Notes*.

2.2.2 Compatibility

You can reuse data base files (*.MDB files) generated by the earlier PROFIBUS Library Editor versions.

The PROFIBUS Library Editor 4.7 upgrades them at their opening.

2.3 Shut-down and Start-up Procedures

Follow the basic shutdown and power up procedure of AS 100ES or your PC as applicable.

2.4 Product Verification

Start the PROFIBUS Library Editor. See Section 3.5, Application Procedures.
Choose the **Help | About …** command to verify product name, version, and copyright notice.

Chapter 3 Configuration/Application Building

3.1 Considerations

3.1.1 Basic Questions

If you want to set up the DB element types or type circuits that are necessary to configure the PROFIBUS-DP link, you need the following information:

1. How shall the network be configured?
   - Network number (bus number)
   - Node numbers (station numbers) of all stations on the PROFIBUS-DP
   - Slaves to be used.
2. What are the line characteristics?

   PROFIBUS-DP bus characteristics
   Specify the speed and the various timing parameters if these require non-standard settings. This might be the case for some special I/O devices. The timings are usually given in units of bit-time, t_{BIT}.
3. Do you want to configure PROFIBUS-DP link and devices only for AC 100 Series systems? Do you want to configure for AC 40 Series and AC 100 Series systems?

   Create a new PROFIBUS database with the PROFIBUS Library Editor in the corresponding *AC 100 only* or *AC 400 and AC 100* preferred mode.
4. Which DB elements do you need in case of AC 100 Series systems?
   Which TCs do you need in case of AC 400 Series systems?

   Create DB element types or TCs for the PROFIBUS-DP slaves by the PROFIBUS Library Editor. You will need the GSD file for each slave to do that.

3.1.2 System Overview

The PROFIBUS Library Editor works differently depending on whether you use it for AC 100 Series or for AC 400 Series systems.

AC 100 Series Usage

For a general understanding of AC 100 Series usage, the following figure shows

- The general data flow within the PROFIBUS Library Editor
- Effect of main PROFIBUS Library Editor commands
- Connection of PROFIBUS Library Editor to Function Chart Builder
- Terminology, such as command names, file names, notions.

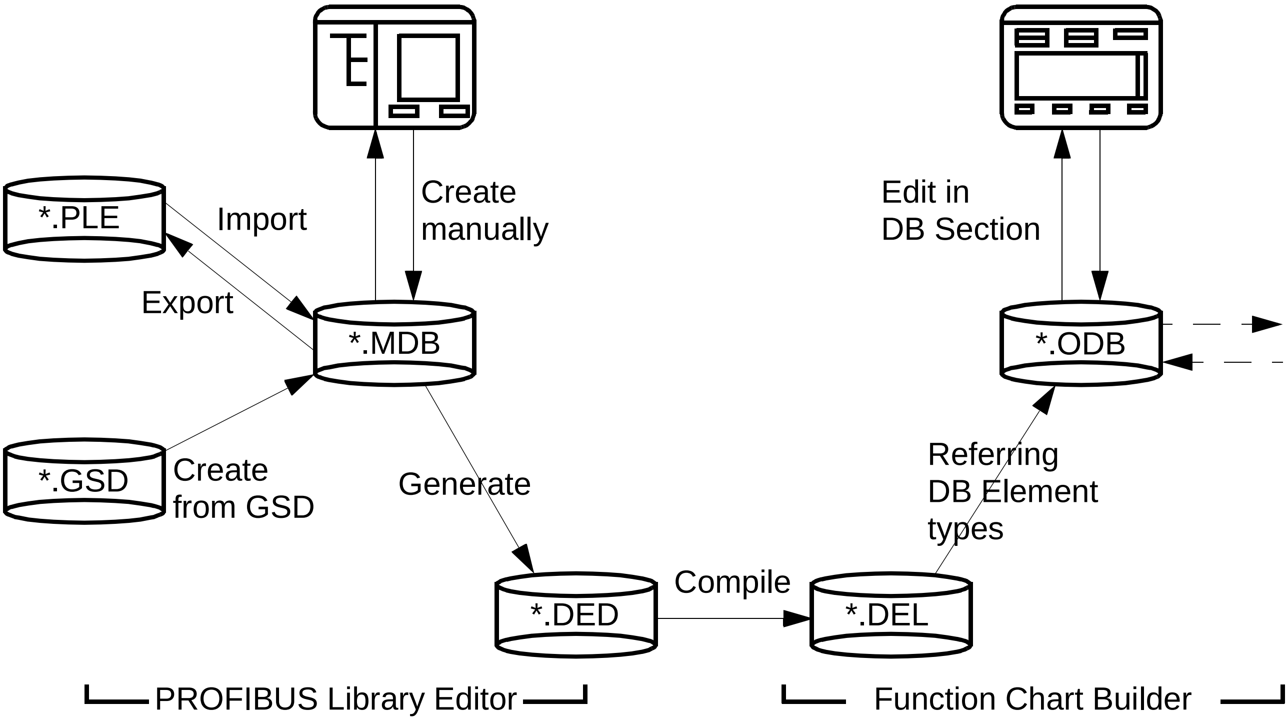


Figure 3-1. PROFIBUS Library Editor for AC 100 Series System, Overview

Different vendors specify their PROFIBUS slaves (and their modules and signals) in Geräte-StammDaten (GSD) files.

National language versions of Geräte-StammDaten files have different extensions, for example, *.GSG for German, *.GSS for Spanish. Nevertheless, we refer to all these files as *.GSD files.

The PROFIBUS Library Editor allows semi-automatic creation of PROFIBUS data base on the basis of *.GSD files. Related command: **Edit | Create**.

*.MDB: The PROFIBUS data base contains the complete description of a set of PROFIBUS slaves you intend to use with your Advant Controller 100 Series stations. It contains—though invisible in the PROFIBUS Library Editor—also the description of the PROFIBUS master implemented by the processor module PM634 with the integrated coupler.

The PROFIBUS Library Editor supports manual entry and modification of PROFIBUS data base of different slaves, modules, and signals. Related command: **Edit | Create**.

Proven PROFIBUS data base of slaves, modules, and signals of different vendors can be stored in the exchange format *.PLE for repeated use. So, you have to create, edit, and test a slave (module, signal) of a certain type only once. Related commands: **File | Export**, **File | Import**.

The *.PLE files build the PROFIBUS library.

For AC 100 Series systems, the PROFIBUS Library Editor generates DB element description source file on the basis of the current PROFIBUS data base. Related command: **Actions | Generate DED for AC 100**.

DB element description source file *.DED pertaining to an AC 100 Series node type contains the DB element types representing the PROFIBUS master and a given set of slaves.

The PROFIBUS Library Editor compiles the DB element description source file to binary DB element description file to be used directly by the Function Chart Builder. Related command: **Actions | Compile DED for AC 100**.

The binary DB element description file *.DEL is a DB element library pertaining to an Advant Controller 100 Series node type. It is assigned to this node type as PROFIBUS DB element library in the FUNCTION.INI file of the Function Chart Builder and is referenced in the target description file *.TEL of this node type.

You represent the PROFIBUS master, slaves, modules, and signals by instances DB elements of the corresponding types.

The *.ODB file is the internal data representation of a node, type circuit, or circuit in the Function Chart Builder.

You create appropriate number (of instances) of DB elements from the DB element types supplied by the DB element libraries. You configure the DB element instances by editing their terminal values in the Function Chart Builder.

The internal data representation is the basis for source code and target code generation.
At source code and target code backtranslation the data are stored again in *.ODB.

AC 400 Series Usage

For a general understanding of AC 400 Series usage, the following figure shows

- The general data flow within the PROFIBUS Library Editor
- Effect of main PROFIBUS Library Editor commands
- Connection of PROFIBUS Library Editor to Function Chart Builder and On-line Builder
- Terminology, such as command names, file names, notions.

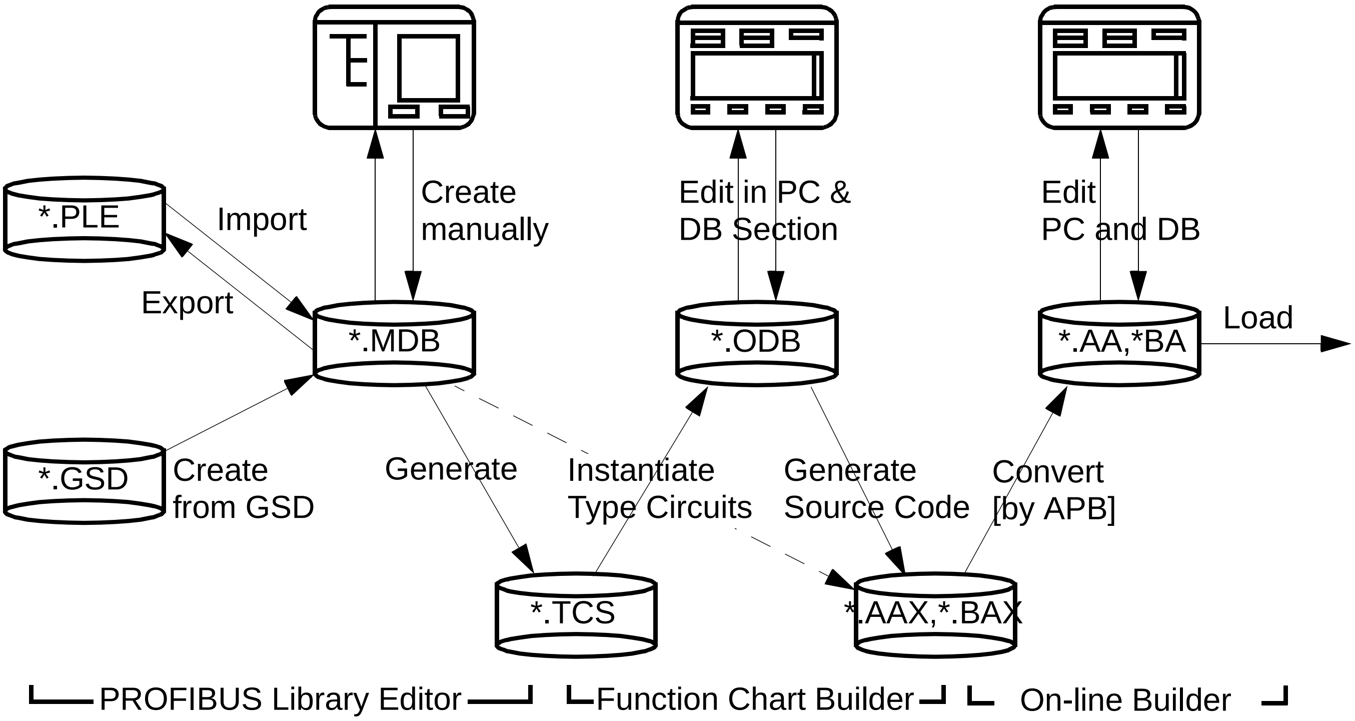


Figure 3-2. PROFIBUS Library Editor for AC 400 Series System, Overview

Explanation for GSD files, PROFIBUS data base creation, *.MDB files, manual entry and modification of PROFIBUS data base, *.PLE files, export, import, and *.ODB file see after Figure 3-1.

For AC 400 Series systems, the PROFIBUS Library Editor generates type circuit source files and AMPL source code template files on the basis of the current PROFIBUS data base. Related command: **Actions | Generate Code for AC 400**.

A set of type circuit source files *.TCS pertain to every PROFIBUS device type to be used in an AC 400 Series system. They contain DB elements to describe a PROFIBUS device type and carry its configuration data, or PC and DB elements to define the address of a device of that type and offer its input, output, and diagnostic terminals.

A pair of *.AAX and *.BAX file will be also generated for every PROFIBUS device type. They contain the same DB and PC elements as the corresponding *.TCS files and can serve as templates for work with the On-line Builder.

In the Function Chart Builder, you declare the PROFIBUS device types by instantiating the corresponding D* type circuits in the DB Section of a node or node structure. Then you instantiate one instance of the corresponding P* type circuits for every PROFIBUS device connected to that node.

Generate source code (*.AAX and *.BAX files) of the application in Function Chart Builder. They contain the complete program and database of the application.

Convert the generated *.AAX and *.BAX files to *.AA and *.BA files with the Application Builder.

Use the On-line Builder to download the converted application to the AC 400 Series system. Alternatively, open the template files with a simple editor and edit the application with the On-line Builder on their basis.

3.1.3 Representations of PROFIBUS Configurations

Physical PROFIBUS Configurations

The PROFIBUS configurations consist of two parts:

- The highly standardized part of PROFIBUS master implemented by the processor module PM634 with the integrated coupler or by the coupler CI541
- The highly variable part of PROFIBUS slaves implemented by different vendors. The latter are defined by their GSD.

PROFIBUS Devices in PROFIBUS Library Editor

In the PROFIBUS Library Editor, PROFIBUS and PROFIBUS device types (master, slaves, module and signal types) of different vendors are represented by the PROFIBUS data base. Excepts of the PROFIBUS data base can be stored in the exchange format *.PLE for repeated use. The *.PLE files make up the PROFIBUS element library. (Do not mix up the latter with a DB element library.)

Advant Controller 100 Series DB Element Libraries

When you work with AMPL Control Configuration on Advant Controller 100 Series nodes, also PROFIBUS and PROFIBUS device types (master, slaves, module and signal types) are represented by DB elements (DB element types). They are collected in DB element libraries. A DB element library is offered for and referenced by a corresponding node type, for example, Advant Controller 110 2.2.

Advant Controller 100 Series PROFIBUS Configurations in Function Chart Builder

In the Function Chart Builder, you configure the DB part of your application by

- Instantiating the appropriate number and kind of DB elements (DB element types) representing the PROFIBUS, the PROFIBUS master, and the slaves with their modules and signals
- Editing their terminal values.

Type Circuit Libraries

When you work with AMPL Control Configuration on Advant Controller 400 Series nodes, PROFIBUS and PROFIBUS device types and instances are represented by PC and DB elements packed in type circuits. The latter are stored either as project-specific TC library in `project_root\project*\NODES\PR_TC\TCDATA` or in `TC$LEV1` or `TC$LEV2` directory for general use. Modular slave types are represented by a set of compact slave types, one for each configuration variant of the modules.

Advant Controller 400 Series PROFIBUS Configurations in Function Chart Builder

- In the DB section you instantiate the appropriate kind of D* type circuits representing the PROFIBUS, the PROFIBUS master, and the slave types with their configuration data.
- In the PC section you instantiate the appropriate kind and number of P* type circuits representing the slaves and their signals.

3.2 Capacity and Performance

The PROFIBUS Library Editor as Windows task can be started several times, that is, several instances of the PROFIBUS Library Editor can run at a time. Though, they must work on different Access databases (different *.MDB files).

3.3 Application Start-up

Start the PROFIBUS Library Editor preferably by the following method:

In the **Start** menu of Windows NT choose

Start | Advant | Engineering | Utilities | PROFIBUS Library Editor 4.7-*

The PROFIBUS Library Editor starts and expects you to open some data base file or create a new one.

3.4 Tutorial

Not applicable.

3.5 Application Procedures

The necessary steps to build the application data base and PC program for PROFIBUS-DP communication are different depending on whether your target is an Advant Controller 100 Series system or an Advant Controller 400 Series system.

The following tables show the usage of the PROFIBUS Library Editor embedded in a larger context. Use the following tables to overview the steps and also as checklists.

This is a checklist of the necessary steps to build the application data base and PC program for the PROFIBUS-DP communication in Advant Controller 100 Series systems.

Table 3-1. Checklist for PROFIBUS-DP Applications in AC 100 Series

No	Issue	OK
1	Define the network number for the PROFIBUS-DP line.	
2	Specify the PROFIBUS-DP bus characteristics, such as speed and the various timing parameters if these require non-standard settings. This might be the case for some special I/O devices.	

Table 3-1. Checklist for PROFIBUS-DP Applications in AC 100 Series

No	Issue	OK
3	Define the number, kind, and type of PROFIBUS-DP slaves to be used.	
4	Get the GSD file for each slave (and module and signal) type.	
5	Define the node numbers for all slaves, that is, their address on the PROFIBUS-DP bus.	
6	Check that the DB element library contains the necessary DB element types. If not, Create a DB element type for each PROFIBUS-DP slave (and module and signal) type by using PROFIBUS Library Editor. Make the new DB element types available for the Function Chart Builder by using Setup Add-Ons.	
7	In Function Chart Builder, instantiate and configure the processor module DB element, for example, PM634.	
8	Check whether any of the slaves needs an unusual timing.	
9	Instantiate and configure DB element PB.	
10	Instantiate and configure slave elements and their subordinates as applicable.	
11	Define the cycle time of your control modules.	
12	Build the control module, program the PC part.	
13	Generate target code and load it to the target system.	
14	Test the application.	

This is a checklist of the necessary steps to build the application data base and PC program for the PROFIBUS-DP communication in Advant Controller 400 Series systems.

Table 3-2. Checklist for PROFIBUS-DP Applications in AC 400 Series

No	Issue	OK
1	Define the network number for the PROFIBUS-DP line.	
2	Specify the PROFIBUS-DP bus characteristics, such as speed and the various timing parameters if these require non-standard settings. This might be the case for some special I/O devices.	
3	Define the number, kind, and type of PROFIBUS-DP slaves to be used.	

Table 3-2. Checklist for PROFIBUS-DP Applications in AC 400 Series

No	Issue	OK
4	Get the GSD file for each slave (and module and signal) type. In case of modular slaves, edit a special GSD file for each slave variant. In each of them you define a variant as a separate compact slave.	
5	Define the node numbers for all slaves, that is, their address on the PROFIBUS-DP bus.	
6	Check that the one of the available TC element library contains the type circuits carrying the necessary slave types (configuration data, address properties, terminals). If not, Create a pair of type circuits for each PROFIBUS-DP (compact) slave type by using PROFIBUS Library Editor. Or make the necessary type circuits available for the Function Chart Builder by copying from other projects or other sources.	
7	Check whether any of the slaves needs an unusual timing	
8	In Function Chart Builder, instantiate and configure DB element CI541	
9	Instantiate the appropriate D* type circuits once in the DB section.	
10	Instantiate and configure the corresponding P* type circuits for every (compact) slave in the PC section.	
11	Define the cycle time of your control modules	
12	Build the control module, program the PC part.	
13	Generate source code)*:AAX and *:BAX files) of the application.	
14	Convert the source code to *:AA and *:BA files by the Application Builder.	
15	Load the *.AA and *.BA files to the target system by the On-line Builder.	
16	Test the application.	

You find the detailed description of the procedures in Section 4.3, Operating Instructions.

Chapter 4 Runtime Operation

4.1 Operating Overview

4.1.1 About Preferred Mode

Implementation of PROFIBUS master and link and representation of PROFIBUS slaves are different in case of Advant Controller 100 Series and Advant Controller 400 Series systems. The PROFIBUS Library Editor supports both systems and offers basically all necessary—and different—functions for both methods.

Nevertheless, when working for AC 400 Series systems, it is convenient to rely on the PROFIBUS Library Editor to ensure limitations:

- Also Modular Slave stations have to be represented as Compact Slave stations.
- A unique maximum 7 character long short name has to be defined.
  The short name with a leading ‘D’ and ‘P’ builds the name of the D* and T* type circuit, respectively.

By the setting the *preferred mode* at creation of a PROFIBUS data base with the PROFIBUS Library Editor you define whether you want to apply these limitations.

So, *AC 100 only* preferred mode means that

- You can (but you need not to) represent modular slaves—even compact slaves—as modular slaves and define modules and slaves when creating a slave by hand.
  The PROFIBUS Library Editor can convert slaves described as modular to compact at automatic slave creation on the basis of *.GSD files.
- You can ignore the short names.

AC 400 and AC 100 preferred mode means that

- You can create only compact slaves by hand, that is, you have to represent also modular slaves as compact ones.
  The PROFIBUS Library Editor converts slaves described as modular to compact at automatically slave creation on the basis of *.GSD files.
- The PROFIBUS Library Editor suggests a unique short name. You can edit it.

The following table shows the relation between module types on device level, as description in *.GSD file, and representation by PROFIBUS Library Editor:

Table 4-1. Compact or Modular?

Hardware	Compact HW construction		Modular HW construction	
\ Modular_ Station= \ in *.GSD System	**0** (described as compact)	**1** (described as modular(1)	**0** (described as compact)(2)	**1** (described as modular)
AC 100 Series	Represented as compact in PLE	Represented as compact or modular(3) in PLE	Represented as compact in PLE	Represented as modular(3) in PLE
AC 400 Series	Represented as compact in PLE	Represented as compact(4) in PLE	Represented as compact in PLE	Represented as compact(4)(5) in PLE

(1) Though the slave has a compact hardware construction, the manufacturer may prefer to describe it as a modular slave to achieve a well-structured description in the *.GSD file.

(2) It is the user who has to edit a “compact” *.GSD file for each occurring configuration of modules. This is normally the case with modular slaves for AC 400 Series systems.

(3) The user has to take care at the instantiation of module and signal DB elements in the Function Chart Builder to ensure conformity with the underlying compact slave device.

(4) Conversion from modular to compact is default and fixed.

(5) It is the user who has to check the “modular” *.GSD file and delete unnecessary module definition variants to facilitate automatic conversion to compact.

4.1.2 Use of PROFIBUS Library Editor for Advant Controller 100 Series

When you work with the PROFIBUS Library Editor for Advant Controller 100 Series systems, you produce PROFIBUS DB element library reflecting actual PROFIBUS configurations.

You set either *AC 100 only* or *AC 400 and AC 100* preferred mode when you create the PROFIBUS data base with the PROFIBUS Library Editor.

1. You produce one or more DB elements, in fact, DB element types, for the PROFIBUS slave stations of AC 100 Series stations depending on slave type.
   - A Compact Slave DB element type is created to represent a PROFIBUS slave device if the device is declared as compact by the manufacturer (`Modular_Station=0` in *.GSD file).
   - A Modular Slave DB element type and a set of Module DB element types are created to represent a PROFIBUS slave device if the device is declared as modular by the manufacturer (`Modular_Station = 1` in *.GSD file) and the process values can be collected into some functional groups, like various I/O belonging to a compact motor control, rather than into identical periodical groups, like 16 identical digital inputs, then 8 digital outputs, and these groups correspond to the modules.
     A Modular Slave DB element type and a set of Module DB element types are created to represent a PROFIBUS slave device also if the slave is defined according to the PROFIBUS-DPV1 standard.

   - A Modular Slave DB element type, a set of Module DB element types, and a set of Signal DB element types are created to represent a PROFIBUS slave device if the device is declared as modular by the manufacturer (`Modular_Station = 1` in *.GSD file) and the process values can be collected into identical periodical groups, like 8 times (analog input value, then status), then 4 analog outputs,
     rather than into some functional groups, like various I/O belonging to valve control, and these groups correspond to the modules.

2. You can use the PROFIBUS Library Editor later to refine or to redefine DB element types. For example, you may want to profit from advantages of extended DB elements or signal referencing by name or symbol offered by Signal DB element types:

   Though a device is defined according to the PROFIBUS-DPV1 standard—hence no Signals have been generated automatically—, but the process values can be collected into identical periodical groups corresponding to the modules, you can manually create one Signal DB element type for each Module DB element type with the PROFIBUS Library Editor.

The DB element types comprise all data necessary to configure and access the PROFIBUS slave stations.

NOTES

The corresponding DB element types have to be generated only once for every PROFIBUS slave type. The DB elements types can be distributed to and used with any installation of the Function Chart Builder.

The PROFIBUS Library Editor 4.7, as well as the whole AMPL Control Configuration, is a 32-bit application running under Windows NT. The DB element types created by the PROFIBUS Library Editor 4.7 cannot be used with the earlier 16-bit Function Chart Builder releases (part of AdvaBuild for Windows) under Windows for Workgroups 3.11.

4.1.3 Use of PROFIBUS Library Editor for Advant Controller 400 Series

When you work with the PROFIBUS Library Editor for Advant Controller 400 Series systems, you produce a type circuit library reflecting actual PROFIBUS configurations.

You set *AC 400 and AC 100* preferred mode when you create the PROFIBUS data base with the PROFIBUS Library Editor.

1. You produce a pair of type circuits for every PROFIBUS slave type or variant:
   - One D* and one P* type circuit for every slave type for a compact slave station
   - One D* and one P* type circuit for every slave variant for a modular slave station.

NOTE

In case of Advant Controller 400 Series, modular slaves are represented always as compact slaves.For this, each occurring configuration of modules has to be described and represented by a fictitious compact slave type. We call them *slave variants*.

2. You can use the PROFIBUS Library Editor later to refine or to redefine D* and P* type circuits (and DB element types).
   For example, you may want to delete some uninteresting diagnostic terminals so that they do not occur in the P* type circuit.

The PC and DB elements packed in the type circuits comprise all data necessary to configure and access the PROFIBUS slave stations.

NOTES

The corresponding type circuits have to be created and generated only once for every PROFIBUS slave type and slave variant. The type circuits can be distributed as type circuit library in source file (*.TCS file) form and used with any installation of the Function Chart Builder.

The type circuits created by the PROFIBUS Library Editor 4.7 (a 32-bit application under Windows NT) can also be used with the earlier 16-bit Function Chart Builder releases (part of AdvaBuild for Windows) under Windows for Workgroups 3.11.

4.1.4 Configuration and Programming in AMPL Control Configuration

AMPL Control Configuration (32-bit applications running under Windows NT) serves for configuring and programming of Advant Open Control Systems and Advant Open Control Products.

NOTE

The earlier AdvaBuild for Windows (16-bit applications) offered similar functionality under Windows for Workgroups 3.11.

AMPL Control Configuration consists of several components, such as Application Builder, Function Chart Builder, and of options, such as AMPL PC and DB Element Libraries to be installed separately.

Generally, data base elements (DB elements) represent different hardware elements and features, such as couplers, busses, and process I/O modules. Process control elements (PC elements) represent elementary and compound process control functions.

You setup your project and manage the objects (that is, nodes, type circuits, and circuits) belonging to a certain node type in the Application Builder:

> For each object, you select here element libraries in form of node type and options reflecting properties and capabilities of the corresponding Advant Controller, in which the application will run. This is the way PROFIBUS devices are represented and offered for working with AMPL Control Configuration on Advant Controller 100 Series nodes.
> See Section 4.1.5, Database of Advant Controller 100 Series with PROFIBUS.

You can make type circuits available for nodes, node structures, and circuits belonging to a certain project in the project directory structure. Or you make them available for several projects in the level1 and level2 type circuit directory.

This is the way PROFIBUS slave types and variants are offered for working with AMPL Control Configuration on Advant Controller 400 Series nodes. See Section 4.1.6, PROFIBUS Type Circuits for Advant Controller 400 Series.

You configure and program your application with the Function Chart Builder:

- You build the data base part of your application by instantiating (that is, inserting one or several instances of) DB elements available in the selected DB element libraries.
- You build the program part by instantiating PC elements available in the selected PC element libraries.
- You instantiate type circuits in the data base and program part of the application. This is the way PROFIBUS devices of Advant Controller 400 Series systems are represented when working with AMPL Control Configuration.

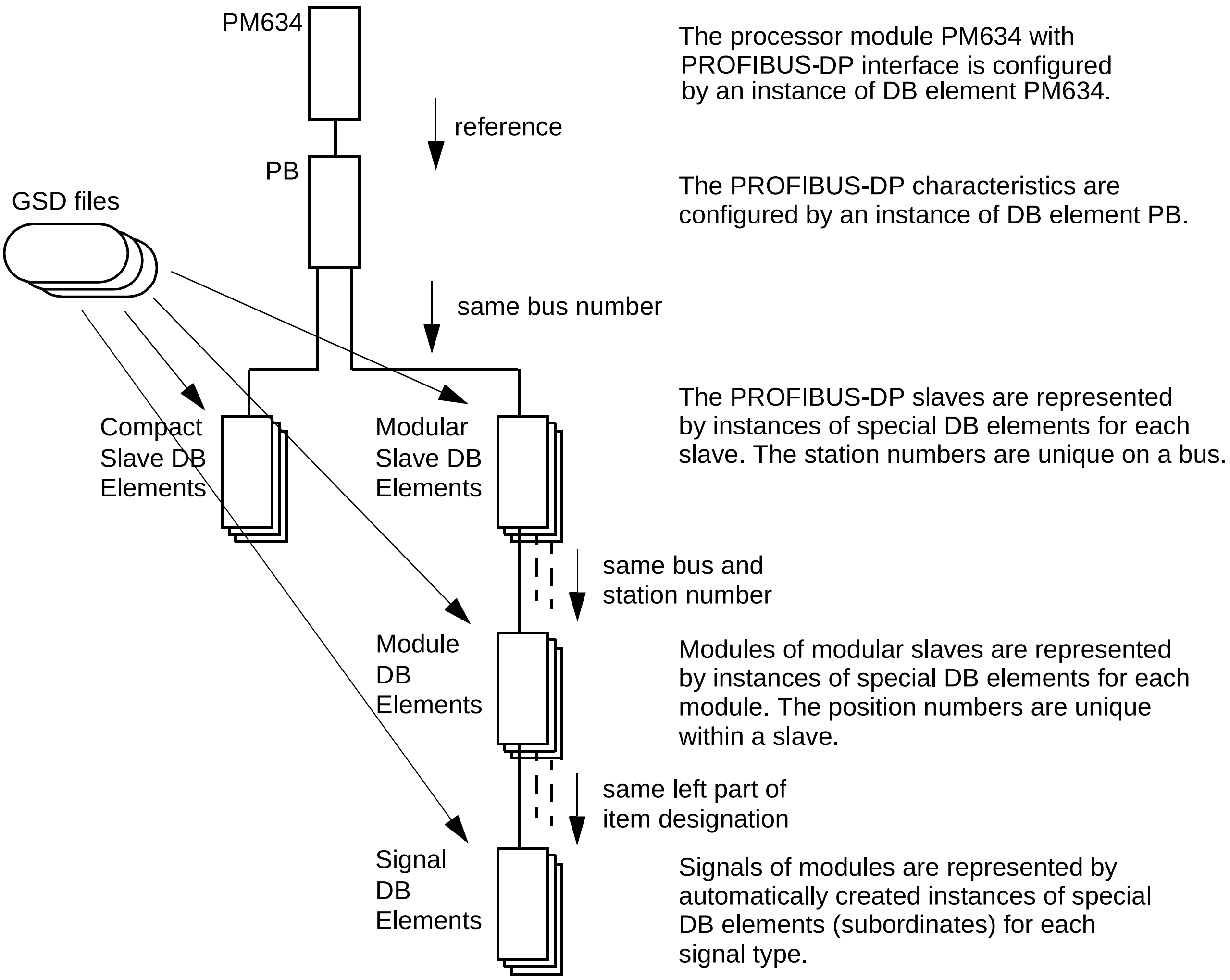
Then, you download the application to the node, start it, and test it. In case of Advant Controller 400 Series nodes you need for this the Application Builder and the On-line Builder.

4.1.5 Database of Advant Controller 100 Series with PROFIBUS

The data base necessary for PROFIBUS-DP configuration in Advant Controller 100 Series applications is defined with the following DB element instances:

- PM*xxx* (for the processor module, such as PM634 for processor module PM634)
- PB (for the PROFIBUS-DP interface and the PROFIBUS master function)
- Slave elements and—in case of modular slaves—their module and signal elements (These device dependent DB element types can be created with the PROFIBUS Library Editor).

In the next figure the hierarchy of these DB elements is shown.


PM634
reference
PB
GSD files
same bus number
Compact Slave DB Elements
Modular Slave DB Elements
same bus and station number
Module DB Elements
same left part of item designation
Signal DB Elements

The processor module PM634 with PROFIBUS-DP interface is configured by an instance of DB element PM634.

The PROFIBUS-DP characteristics are configured by an instance of DB element PB.

The PROFIBUS-DP slaves are represented by instances of special DB elements for each slave. The station numbers are unique on a bus.

Modules of modular slaves are represented by instances of special DB elements for each module. The position numbers are unique within a slave.

Signals of modules are represented by automatically created instances of special DB elements (subordinates) for each signal type.

Figure 4-1. Hierarchy of DB Elements for PROFIBUS-DP Configurations

There are no special PC elements necessary for PROFIBUS-DP configuration in Advant Controller 100 Series applications.

4.1.6 PROFIBUS Type Circuits for Advant Controller 400 Series

Database of Advant Controller 400 Series Application with PROFIBUS

The data base necessary for PROFIBUS-DP configuration is defined with the following DB element instances:

- PM5## (processor module is always available as system DB element)
- CI541 specifies the Communication Interface submodule CI541 supporting PROFIBUS-DP.
- PBSD (PBS Description) specifies configuration and parameters for one or several PROFIBUS slave type and variant.
- PBS (PB Slave) specifies a PROFIBUS slave on a PROFIBUS-DP bus connected to an Advant Controller 400 Series. It contains configuration and diagnostic information of a PROFIBUS slave. Each PBS data base element references a PBSD element.

Program of Advant Controller 400 Series Application with PROFIBUS

The PC program for PROFIBUS-DP representation consists of the instances of:

- PB-R (PROFIBUS - Receive) is used for access of input data from a PROFIBUS-DP slave.
- PB-S (PROFIBUS - Send) is used for sending data to PROFIBUS slave.
- PB-DIAG (PROFIBUS - Diagnostic) is used for access of diagnostic data from a PROFIBUS-DP slave.

They use the PBS DB element to get address information.

The next figure shows the hierarchy of these DB elements packed in type circuits.

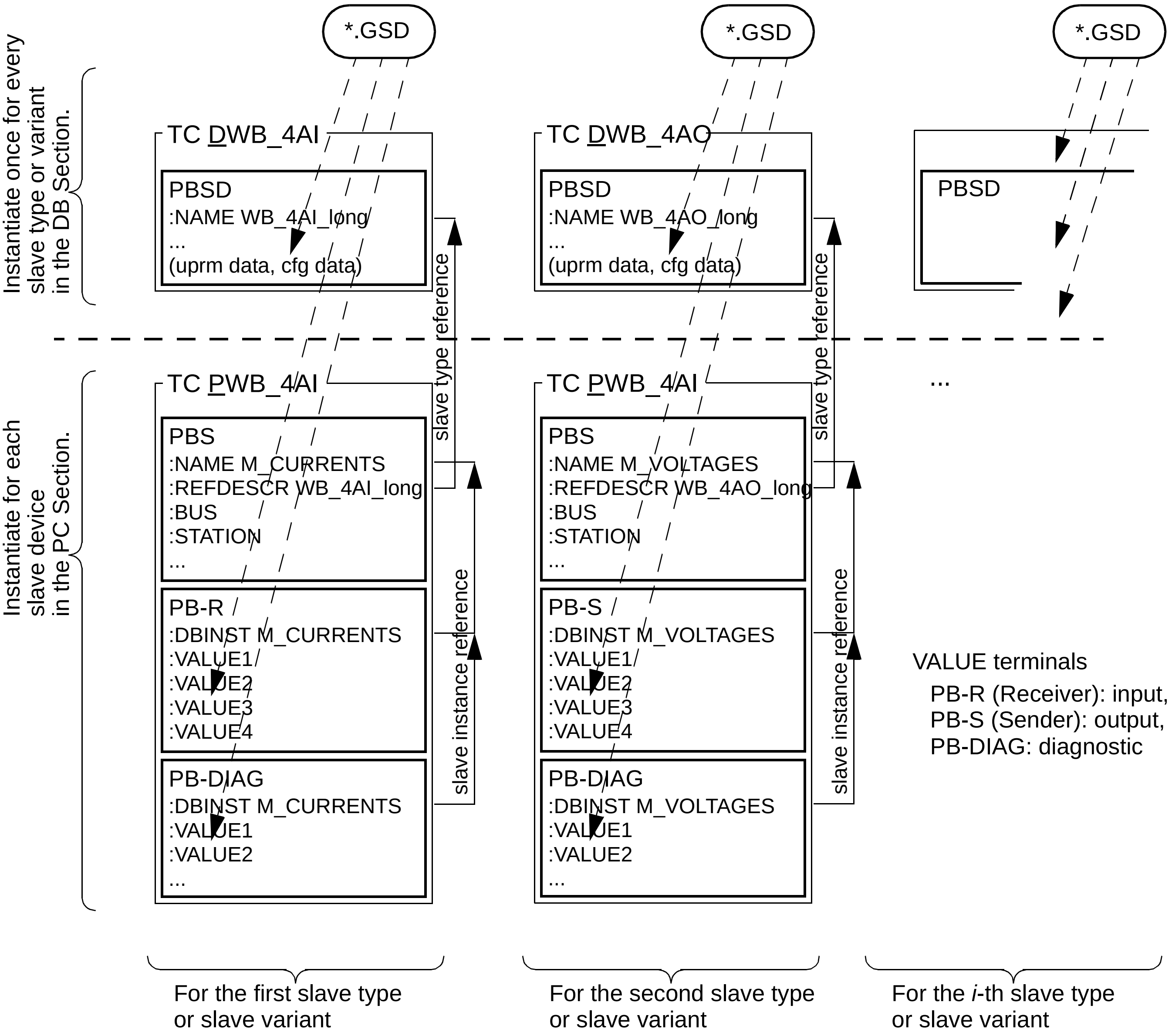


Figure 4-2. Hierarchy of Type Circuits for PROFIBUS-DP Configurations

*.GSD Files

The GeräteStammDaten files contain the user parameter data, configuration data, and information about the number and kind of terminals.

In case of modular slaves, you have to edit new, own *.GSD files for every slave variant (module configuration) that occur in your system. These files describe the slave variants as a set of compact slaves.

D* Type Circuits

D* type circuits contain a PBSD DB element and represent the PROFIBUS, the PROFIBUS master, and the slave types with their configuration data.
* stands for the short name of the slave type, it is maximum 7 characters long.
You instantiate the D* type circuits in the DB section before instantiating the corresponding P* type circuit. The value of NAME terminal is the type of the slave, it is the long name consisting of maximum 20 characters.

P* Type Circuits

P* type circuits contain

- One PBS DB element
  - Referencing the slave type as defined by the corresponding D* type circuit
  - Carrying the address of the slave device
  - Carrying a name
- One or several PB-R (Receiver) PC elements
  - Referencing the “parent” PBS element
  - Carrying the input terminals of the slave device
- One or several PB-S (Sender) PC elements
  - Referencing the “parent” PBS element
  - Carrying the output terminals of the slave device
- One or several PB-DIAG PC elements
  - Referencing the “parent” PBS element
  - Carrying the diagnostic terminals of the slave device.

* stands for the short name of the slave type, it is maximum 7 characters long.

Number and Identity of P* Instances

The number of PB-# elements is derived from the configuration data contained in the corresponding D* type circuit. You instantiate the P* type circuits in the PC section. You instantiate a P* type circuit as many times as many devices of this type and variant are connected to the given AC 400 Series controller.

Symbols of P* Type Circuit

PBS_NAME

> The value of PBS_NAME is the (DB) instance name of the PBS DB element contained in the P* type circuit. You assign a unique value, for example, M_CURRENTS, at the instantiation of the P* type circuit.

BUS, STATION

> The value of BUS, STATION is the unique address for the PBS DB element contained in the P* type circuit.

__INAM__

> This is a default—and in the **Edit Symbol Values** dialog box of Function Chart Builder invisible—symbol. It carries the instance name of the PBS DB element (<PBS_NAME>) from within the type circuit to outside. The Function Chart Builder assigns the value of __INAM__ automatically as instance name to the type circuit instance at its instantiation.

NOTE

This feature allows very similar connection of PROFIBUS_DP signals in the PC section for both AC 400 Series and AC 100 Series systems.

Example of PC source code in AC 400 system:

`PC1.1.18 MOVE (B,1)`
`:1 ABCD:IN1`; PROFIBUS-DP device is represented by type circuit ABCD

Example of PC source code in AC 100 system::

`PC1.1.18 MOVE (B,1)`
`:1 =ABCD:IN1`; PROFIBUS-DP device is represented by DB element ABCD

4.2 Runtime Tutorial

Not applicable.

4.3 Operating Instructions

For a description of commands for control of the PC program and Database, please refer to *AMPL Configuration AC 100 Series Reference Manual*.

4.3.1 Preparation, Check of Installation and Files

1. It is supposed that AMPL Control Configuration is installed on your AS 100ES or personal computer. For installation see *AMPL Control Configuration Release Notes*.

   The PROFIBUS Library Editor and the model element library are attached to Function Chart Builder, you do not need to install them separately.

   The executable is installed in the release directory of the Function Chart Builder

   `drive:\AdvaBNT\Fcb\release_x\BIN32`

   where *drive* is **C**, **D**, and so on, and *release_x* is the subdirectory name of the FCB release, for example, `47r0` standing for FCB*4.7/0.

   The model data base PROFIBUS.MDB is stored in the directory

   `drive:\AdvaBNT\Fcb\release_x\BIN32\db_lib\source`

2. Be sure that the AMPL DB and PC element libraries of the Advant Controller 100 Series or Advant Controller 400 Series or both that you intend to configure and program are installed.

   You can check that in the Application Builder. The corresponding node type, for example, AC 110 2.2, must be available. (The support of PROFIBUS on the target system is normally covered by the base software and not by some option.)

   In due case install the AMPL DB and PC Element Libraries 1.2 or higher for Advant Controller 110. See *AMPL DB and PC Element Libraries Release Notes*.

3. Microsoft's Data Access Objects (DAO) 3.5 is used to work with the PROFIBUS data base files. Be sure it is installed.

   You can check it by choosing **Start | Settings | Control Pane**l of Windows NT and selecting **Add/Remove Programs** and the **Install/Uninstall** tab.
   Though *.MDB files are Microsoft Access files, Microsoft Access 97 need not to be installed on your AS 100ES or personal computer.

In due case install DAO 3.5. See *AMPL Control Configuration Release Notes*.

4. In case of Advant Controller 100 Series, the PROFIBUS Library Editor saves the generated data base files and DB element description files in a user defined directory, as default, in C:\PB_LIB. This directory can be changed by setting the environment variable PB$LIB.
You can check the value of PB$LIB—if PB$LIB is defined at all—by choosing **Start | Settings | Control Panel** of Windows NT and selecting **System** and the **Environment** tab.
If you want to use another folder for PROFIBUS data bases, enter PB$LIB in the **Variable** text box of and the full path of the folder, for example, D:\PB_LIB, in the **Value** box, and press **OK**.

   In case of Advant Controller 400 Series, the PROFIBUS Library Editor saves the generated type circuit files in the project-specific type circuit directory or in level1 or level2 type circuit directories. The latter are, as default, in C:\TCLEV1 and C:\TCLEV2 These directories can be changed by setting the environment variables TC$LEV1 and TC$LEV2.

   You can check and set the value of TC$LEV* in the same way as that of PB$LIB.

4.3.2 Editing GSD of PROFIBUS Devices and Variants

1. Study the documentation of the PROFIBUS slave devices you want to work with and their *.GSD file.
Find out which standard the *.GSD files follow: PROFIBUS-DP or PROFIBUS-DPV1 (`DPV1_Slave=1`).
Find out whether the device is described as compact (`Modular_Station=0`) or modular (`Modular_Station=1`) by the manufacturer.

NOTE

A modular device may be described as compact or a set of compact types and a compact device may be described as a modular type.

2. For each device, find out its structure and its modularity, which values are transferred via the PROFIBUS to and from the device, what data types the values have, and whether the values can be collected into identical periodical groups.
3. Consider the project environment, in which you intend to work with the DB element types to be created:
   - Is it expected or necessary to reference process signals by instance names or symbolic instance names when working in Function Chart Builder?
If yes, prefer modular description and set *AC 100 only* preferred mode at PROFIBUS database creation.
   - Do you want or have to use extended DB elements in conjunction with PROFIBUS devices?
If yes, prefer modular description and set *AC 100 only* preferred mode at PROFIBUS database creation.

   These questions are especially important when you adapt existing projects to PROFIBUS devices.

- Do you intend to use PROFIBUS devices in AC 400 Series systems or also in AC 400 Series systems
  If yes, set *AC 400 and AC 100* preferred mode at PROFIBUS database creation. Be aware that you will work anyway with a compact representation: either you have already compact description in *.GSD file, or you will convert from modular to compact by the PROFIBUS Library Editor.

4. Prepare appropriate *.GSD files. Use a simple editor, like AS 100-EDIT or Notepad.
   - Case of *.GSD file of compact type containing modules

     Some manufacturer, such as Softing and Bosch, use the modules for attachment of texts. Others, such as Siemens, use them to describe variants of the device.
     In this second case, for each variant that has to be used:

     a. Prepare a backup copy of the original *.GSD file.

     b. Rename the *.GSD file, for example, by including the name of the variant in the filename.

     c. Rename the value of `Model_Name` in *.GSD file, for example, by including the name of the variant.

     d. Remove all module definitions except the used variant from the *.GSD file.

   - Case of Advant Controller 400 Series and modular slaves

     Consider which variants (module configurations of the modular slaves) occur in your system. Create a set of *.GSD files each describing one variant type:

     a. Prepare a backup copy of the original *.GSD file.

     b. Rename the *.GSD file, for example, by including the name of the variant in the filename.

     c. Rename the value of `Model_Name` in *.GSD file, for example, by including the name of the variant.

     d. You can change the slave type to compact (`Modular_Station=0`).
     If you leave the slave type modular (`Modular_Station=1`) and you start creating the PROFIBUS database in *AC 400 and AC 100* preferred mode, the PROFIBUS Library Editor converts the modular description anyway to compact representation at data base creation on the basis of the *.GSD file.

     e. Reedit the *.GSD file so that only those module definitions remain that are constituents of the given variant, and so many times and in an order that corresponds to the configuration of the given variant and ensures the successful conversion to compact representation by the PROFIBUS Library Editor.

4.3.3 Before Starting the PROFIBUS Library Editor

1. If you prepared objects, that is, nodes, node structures, circuits, and type circuits, with Function Chart Builder that use PROFIBUS devices, save your work in form of source code (generate source code).

CAUTION

Later, in step 3 of the phase Section 4.3.9, Working with the PROFIBUS DB Element Library for AC 100 Series you will have to initialize the objects.
If you have not saved your work in form of source code, you will lose it!

2. Exit the Function Chart Builder.
   You should not start the PROFIBUS Library Editor while the Function Chart Builder is running. You should not start the Function Chart Builder while the PROFIBUS Library Editor is running.

4.3.4 Starting the PROFIBUS Library Editor

Start the PROFIBUS Library Editor preferably by the following method:

In the Start menu of Windows NT choose

Start | Advant | Engineering | Utilities | PROFIBUS Library Editor 4.7-*

The PROFIBUS Library Editor starts and expects you to open some data base file or create a new one.

4.3.5 Editing the PROFIBUS Library

1. Create a new data base by **File | New** command. Set preferred mode for it. The PROFIBUS Library Editor copies the PROFIBUS.MDB model data base to C:\PB_LIB\NewDB*nn*. Or, open an existing data base by the **File | Open** command. The original file is preserved until you save your work by the **File | Save** command or exit with save.

   Your current data base contains the description of the PROFIBUS and the PROFIBUS master, which will be represented later by the PB DB element. The PROFIBUS and the PROFIBUS master is not visible in the PROFIBUS Library Editor.

You have alternatives to complete and edit the data base: see sequences 2.a, 2.b, and 2.c below. See also notes to control elements in Section 4.4.1, Controls of PROFIBUS Library Editor.

2.a Filling the data base manually

- a.1 Open the *.GSD file of the slave device or variant by some editor, such as AS100-EDIT, parallel with the PROFIBUS Library Editor.
  You can copy and paste the items between the two applications.
- a.2 Create new elements by the **Edit | Create** command.
  Unselect the **From GSD option**, select the element type, and define its name in the **Create** dialog box. Press **OK**.
- a.3 Fill the different empty fields **Name**, **Description**, and so on in the right pane.
  Contents of a field are accepted at the moment when you select another field.
  The created element appears in the left pane, its properties appear in the right pane.
- a.4 Fill the fields in the different tabs as applicable.

  You can fill the standard terminals by assigning default values:
  Name, **Default Value**, and so on.

  You have to fill the process terminals—possibly by copying and pasting from the *.GSD file: **Position**, **Name**, **Data Type Conversion**, and so on.

2.b Filling the data base semi-automatically on the basis of the *.GSD files of the corresponding slave devices or variants

b.1 Create new elements by activating the **Edit | Create** command.
Select the **From GSD** option and browse for the corresponding *.GSD file in the **Create** dialog box. Press **OK**.
The *.GSD file contains the element type and name. You can check it in the **Preview** window.

b.2 Select the **Convert Slave described as modular to compact** check box in the Preview window if necessary.

b.2 PROFIBUS Library Editor checks the *.GSD file and presents the future DB element types in the **Preview** window. You can accept or rename them.
PROFIBUS Library Editor maps the *.GSD file to DB element terminals.
See Appendix A, Data Interpretation by PROFIBUS Library Editor.

b.3 Not all the fields can be completed automatically. So check the fields and complete the values in the different tabs as applicable. See step a.4 above.

2.c Filling the data base by import from the PROFIBUS element library.

c.1 Choose the **File | Import** command. Select the proven *.PLE file that contains one or several element types you require.

c.2 Check the fields and complete the values in the different tabs of the imported elements as applicable. See step a.4 above.

During these steps in the right pane, you can delete a terminal, duplicate a terminal, delete a window, and delete a link by activating the corresponding button. During these steps in the right pane, you can delete the selected element by the **Edit | Delete** command.

3. Check the current data base fully by choosing the **Actions | Check** command to avoid certain errors later during compilation of *.DED file.

4. Save your work by the **File | Save** or the **File | Save As** command.

4.3.6 Compiling the PROFIBUS Library for AC 100 Series

1. To generate a new DB element description source file (*.DED file) from the current data base, activate the **Actions | Generate DED for AC 100** command. Select the node type, for which you prepared the current data base.

   The PROFIBUS Library Editor sets the name of the *.DED file for that node type, for example, QAC16122.DED on the basis of the [PROFIBUS] group in the FUNCTION.INI file of the Function Chart Builder.

   The PROFIBUS Library Editor checks the current data base partly and generates a *.DED file, for example, QAC16122.DED, from it in the `drive:\AdvaBNT\Fcb\release_x\BIN32\db_lib\source` directory.

2. To compile the new DB element description source files to a binary DB element description files and to check their appropriate referencing from the *.TEL target description files, activate the **Actions | Compile DED for AC 100** command.

The PROFIBUS Library Editor compiles, for example, the QAC16122.DED file and moves the result files, in this case QAC16122.DEL and QAC16122.DCL, to the `drive:\AdvaBNT\Fcb\release_x\BIN32\db_lib` directory.

In case of errors, the PROFIBUS Library Editor generates a *.LST file, for example, QAC16122.LST, in this directory. Open it with by some editor, such as AS100-EDIT, to see the errors found in the corresponding *.DED file.

The PROFIBUS Library Editor checks the *.TEL files in the release directory and ensures the referencing of the new DB element descriptions by AC 110 2.2 node type.

4.3.7 Generating Type Circuits and Template for AC 400 Series

You can generate TCs and template files for Advant Controller 400 Series systems if the current database has been created in *AC 400 and AC 100* preferred mode (and not in *AC 100 only* preferred mode).

1. To generate a new type circuit source files and template files from the current data base, activate the **Actions | Generate Code for AC 400** command.

   The PROFIBUS Library Editor sets the name of the D*.TCS and P*.TCS type circuit source file pair on the basis of the short name and the name of the *.AAX and *.BAX template file pair on the basis of the current *.MDB database name.

2. In the **Generate Code for AC 400** dialog box, select the destination for the D*.TCS and P*.TCS file. Select a project-specific type circuit directory `project_root\project*\NODES\PR_TC\TCDATA` if you want to use the generated type circuits only within that project.
   Select TC$LEV1 or TC$LEV2 for general use.

   The PROFIBUS Library Editor checks the current data base partly and generates the D*short_name*.TCS and P*short_name*.TCS type circuit source file pair in the selected folder.
   It generates also the *current_PROFIBUS_database_name*.AAX and *current_PROFIBUS_database_name*.BAX template file pair in the directory shown by the environment variable PB$LIB (default `C:\PB_LIB`).

4.3.8 Exiting the PROFIBUS Library Editor

Exit the PROFIBUS Library Editor by the **File | Exit** command.

4.3.9 Working with the PROFIBUS DB Element Library for AC 100 Series

1. Select the Advant Controller 100 Series object you are going to work on in the Application Builder.
2. Start the Function Chart Builder from the Application Builder for this object.
3. Initialize the object to make the new DB element library supporting PROFIBUS effective if the Function Chart Builder prompts you to do so. Else go to step 5.

CAUTION

Be sure saved your previous work in Function Chart Builder concerning the different objects in form of source code in step 1 of phase Section 4.3.3, Before Starting the PROFIBUS Library Editor. Else you loose it at initialization and cannot restore it by backtranslation in the following step!

4. Backtranslate your former work if applicable.
5. Configure and program your application consisting of an Advant Controller 100 Series controller with a PROFIBUS interface and PROFIBUS slave stations connected to a PROFIBUS:
   - In the DB section create an instance of a PROFIBUS Master DB element and fill in the correct values at the terminals on the left side of the DB element if required.
   - For every PROFIBUS compact slave station create an instance of a Compact Slave station DB element and fill in the correct values at the terminals on the left side of the DB element if required.
   - For every PROFIBUS modular slave station create an instance of a Modular Slave station DB element and fill in the correct values at the terminals on the left side of the DB element if required.
   - For every module of PROFIBUS modular slave stations create an instance of a PROFIBUS Module DB element and fill in the correct values at the terminals on the left side of the DB elements.
     Depending on the module, subordinated Signal elements are created together with the PROFIBUS Module DB element. Fill in the correct values at the terminals on the left side of the Signal DB element.

NOTE

This procedure is analogous to the configuration of S800 remote I/O, where instances of S800 station and S800 module DB elements have to be created.

   - In the PC section connect the process terminals of these DB element instances to PC elements according to the logic of your application.
   - Debug, test, and run the application on the Advant Controller 100 Series targets.
6. Save proven DB element types—one by one or as a group—for later reuse with the PROFIBUS Library Editor by the **File | Export** command.

4.3.10 Working with AC 400 Series Type Circuits Generated by PLE

1. Create or select the Advant Controller 400 Series node, node structure or circuit you are going to work on in the Application Builder.
2. Start the Function Chart Builder from the Application Builder for this object. With the Function Chart Builder edit other parts of the data base and the PC programs that are not directly related to the type circuits generated by the PROFIBUS Library Editor if applicable.

3. In the DB section create one instance of a D* type circuit for each slave type to be served by the current object:

   Select the **DB Section** window in the Function Chart Builder. Choose the **Edit | Create** command. Select **TC** as element type. The Function Chart Builder lists all suitable type circuits. Select the appropriate D* type circuit and activate **OK**.

4. In the PC section create as many instances of the corresponding P* type circuit as many devices are to be served by the current object:

   Select the **PC Section** window in the Function Chart Builder. If you are working on a node or node structure, create missing PCPGM and CONTRM elements and select the appropriate CONTRM.

   Choose the **Edit | Create** command. Select **TC** as element type. The Function Chart Builder lists all suitable type circuits. Select the appropriate P* type circuit. Activate the **Symbol Values** button.

   Fill the values of
   PBS_NAME: instance name, for example, `M_Currents`
   BUS and STATION: address of the slave on the PROFIBUS, for example, `3` and `7`.

   Activate **OK** in the **Edit Symbol Values** dialog box, activate **OK** or **Apply** in the **Create Element** dialog box.

5. Generate source code of the object.

6. If it is a circuit that contains the D* and P* type circuit instances, locate the circuit in a node structure containing other parts of your application.

   If you have a node structure containing the D* and P* type circuit instances directly or packed in circuits, generate node structure source code. Exit Function Chart Builder.

   Start Function Chart Builder for the same node (in node mode). Expand node data and generate source code.

7. Exit the Function Chart Builder.

8. In the Application Builder choose the **File | Convert** command to convert the generated *.AAX, *.BAX, *.TCS, *.TIX and *.TBX files from the general representation to ABB Master representation, that is, to *.AA, *.BA, *.CF, *.TC, *.TI and *.TB files.

   Select **All Source Code Files** in the **Convert Source Code Files - *node_name*** dialog box. Activate **OK**.

9. Start the On-line Builder. Connect the Advant Controller 400 Series target.

   Load, debug, test, and run the application.

   In due case exit On-line Builder and repeat from step 2 to modify the application.

   See *On-line Builder Engineering Stations User’s Guide*.

4.4 Runtime Operation Menus

4.4.1 Controls of PROFIBUS Library Editor

The controls of PROFIBUS Library Editor are available in its application window consisting of two panes, see Section 4.4.1.1, Left Pane of Application Window and Section 4.4.1.2, Right Pane of Application Window.

The commands are available in the menus selectable from the **Menu Bar** and in the context menus, see Section 4.4.2, Commands of PROFIBUS Library Editor, Section 4.4.1.3, Context Menu of Left Pane, and Section 4.4.1.4, Context Menu of Right Pane.

The preferred mode of the current PROFIBUS database is shown in the last field of the **Status Bar**.

4.4.1.1 Left Pane of Application Window

Contains

- Structure Tree of Slaves
- Module Pool
- Signal Pool.

Select here an element to show and edit its properties in the right pane.

Click with the right mouse button in this pane to pop up the context menu with pertaining commands.

You can modify the names of elements (DB element types) only in this pane.

Structure Tree of Slaves

Shows the structure tree of the slaves, their subordinate modules, and signals contained in the current data base.

Select an element here.

The PROFIBUS Library Editor shows its data in the right pane.

The commands concern the selected element.

If you modify a module or a signal, the change is reflected also in the module pool, and signal pool, respectively.

Module Pool

Shows all modules contained in the current data base.

Select an element here.

The PROFIBUS Library Editor shows its data in the right pane.

The commands concern the selected element.

If you modify a module, the change is reflected also in the structure tree.

Signal Pool

Shows all signals contained in the current data base.

Select an element here.

The PROFIBUS Library Editor shows its data in the right pane.

The commands concern the selected element.

If you modify a signal, the change is reflected also in the structure tree.

4.4.1.2 Right Pane of Application Window

Contains the properties of the element selected the left pane.

Select a tab to inspect and edit the corresponding group of properties.

The number and kinds of columns depend on the tab.

Click with the right mouse button in this pane to pop up the context menu with pertaining commands.

A new line is added automatically if you enter a value in the last line.

If you edit in a certain text field, the change is accepted, becomes valid, and is saved at the moment you select another field.

The common control elements in the upper part of the right pane are the following:

Element Type

Compact Slave, Modular Slave, Module, or Signal selected in the left pane.

Name

The callname of the DB element type selected in the left pane.

It is derived from the *.GSD file:

- In *AC 100 only* preferred mode the name is the `Model_Name` value. It is maximum 34 character long.
- In *AC 400 and AC 100* preferred mode the name is truncated to 20 characters. In due case, the uniqueness of names is ensured by replacing the last character or last characters by an index 0, 1,….

When assigning callnames, see also Section A.3.1, Allowed Characters for DB Element Callnames and Short Names.

Short Name

The short (call)name of the DB element type selected in the left pane.

This field is important only in *AC 400 and AC 100* preferred mode.

The short name, like the callname, is derived from the *.GSD file:

> The `Model_Name` value is truncated to 7 characters to build the callnames of D* and P* type circuits and the names of their source files by adding a leading D and P, respectively. In due case, the uniqueness of short names is ensured by replacing the last character or last characters by an index 0, 1,….

When assigning short names, see also Section A.3.1, Allowed Characters for DB Element Callnames and Short Names.

Description

Edit here the description of the DB element type selected in the left pane.

When you instantiate the DB element, the Function Chart Builder displays the description as comment in the left upper corner of the **Edit DB Terminal Values** dialog.

Maximum length of description is 20 characters.

When assigning description, see also Section A.3.3, Allowed Characters for Description of DB Element Type.

Common control element of slaves in the upper part of the right pane is the following:

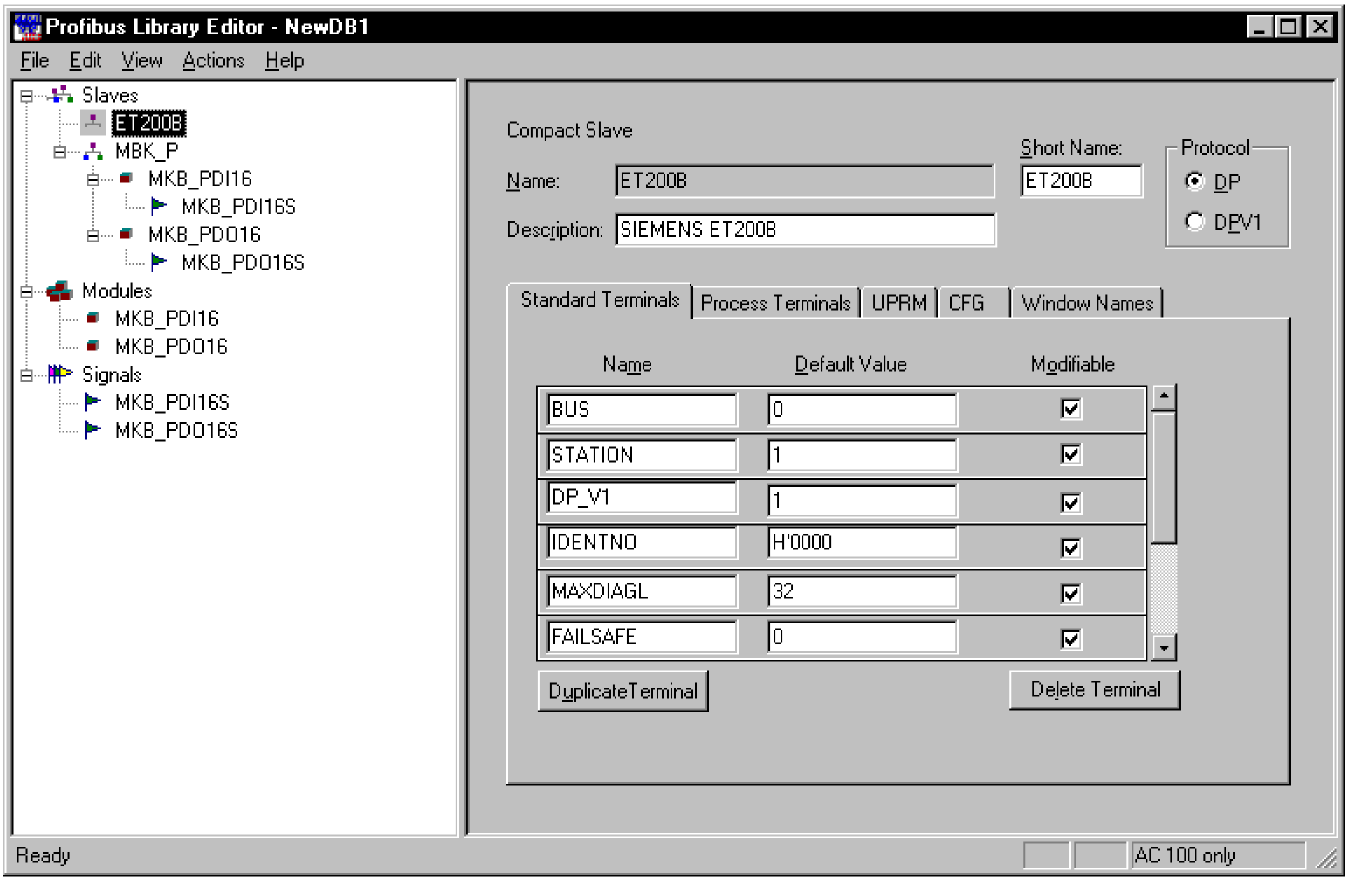
Protocol

Select protocol type here:

- DP if the slave device is defined on the basis of the PROFIBUS-DP standard
- DPV1 if the slave device is defined on the basis of the PROFIBUS-DPV1 standard.

In the following we show figures of the application window. We present the right pane at least once for each element type and once for each tab (but not all of the possible combinations).

Compact Slave, Standard Terminals Tab



Name: ET200B
Description: SIEMENS ET200B
Short Name: ET200B
Protocol: ● DP ○ DPV1

Name	Default Value	Modifiable
BUS	0	☑
STATION	1	☑
DP_V1	1	☑
IDENTNO	H'0000	☑
MAXDIAGL	32	☑
FAILSAFE	0	☑

Figure 4-3. Compact Slave, Standard Terminals Tab

The **Standard Terminals** tab has the same structure for Compact Slave, Modular Slave, and Module.

When working with the Function Chart Builder:

- In case of an AC 100 Series system, standard terminals appear on the left side of the DB element box
- In case of an AC 400 Series system, only standard terminals BUS and STATION are accessible as symbols of the P* type circuit to be assigned at instantiation.

  Name

  Edit the name of terminals here. It can be 1 to 8 characters long. Follow the syntax of DB terminal names, see Section A.3.2, Allowed Characters for DB Terminal Names.

Default Value

Edit the default value of terminals here. They can be part of the *.GSD file. If so, they will be proposed in this field. The value must be in the range allowed for the given terminal, see Section A.4, Standard Terminals of Slave Element Types or Section A.5, Standard Terminals of Module Element Types.

Modifiable

Select this check box to allow modification of terminal values of the corresponding DB element instances in Function Chart Builder. Unselect this check box to fix the terminal values for the corresponding DB element instances.

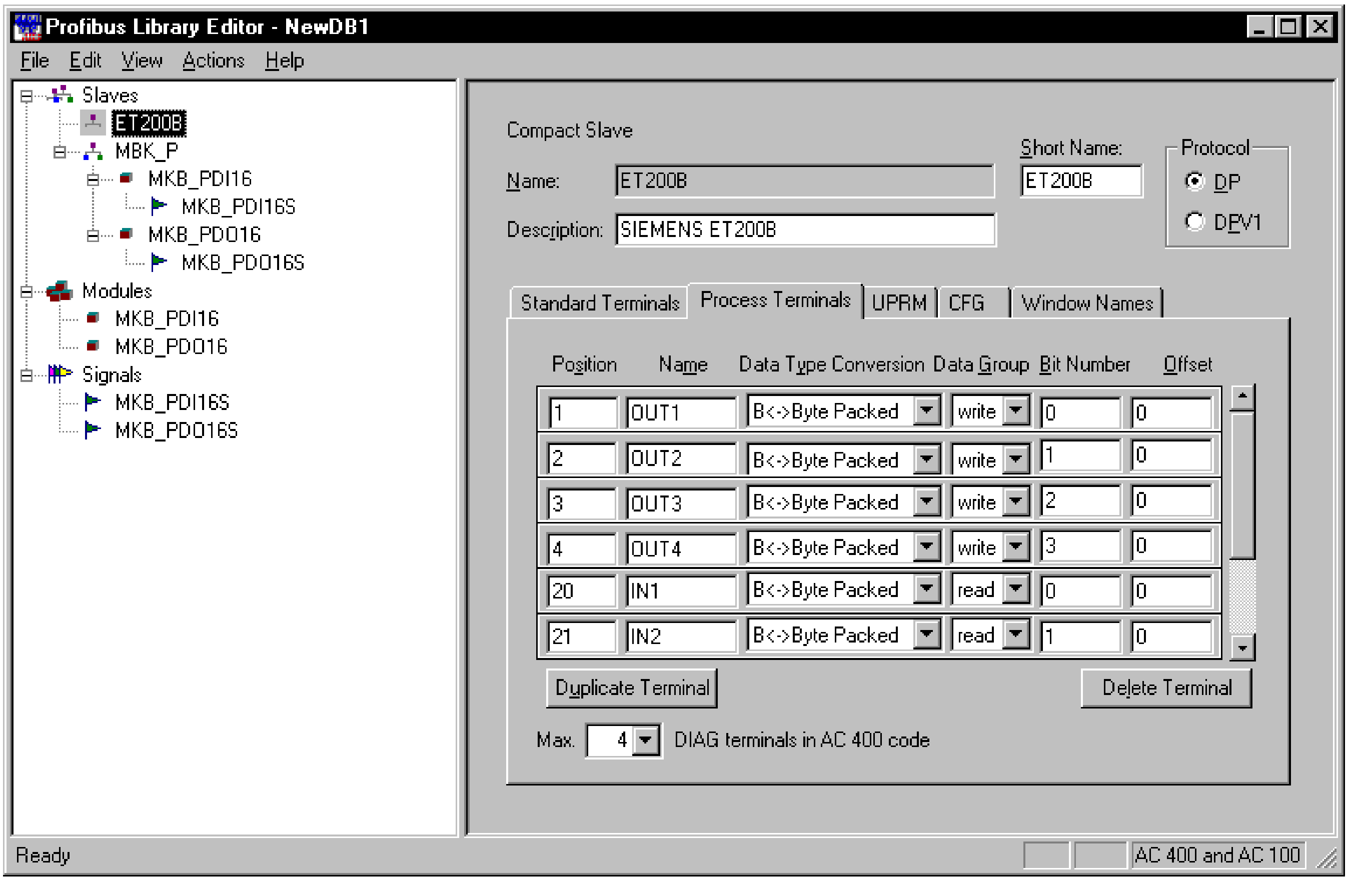
Duplicate Terminal

Activate this button to duplicate the selected terminal and insert it at the last position.

Delete Terminal

Activate this button to delete the selected terminal.
Some terminals can not be deleted.
Some terminals can be deleted, nevertheless they appear in the generated DB element type.

Compact Slave, Process Terminals Tab



Name: ET200B

Short Name: ET200B

Description: SIEMENS ET200B

Protocol: ● DP ○ DPV1

Position	Name	Data Type Conversion	Data Group	Bit Number	Offset
1	OUT1	B<->Byte Packed	write	0	0
2	OUT2	B<->Byte Packed	write	1	0
3	OUT3	B<->Byte Packed	write	2	0
4	OUT4	B<->Byte Packed	write	3	0
20	IN1	B<->Byte Packed	read	0	0
21	IN2	B<->Byte Packed	read	1	0

Max. 4 DIAG terminals in AC 400 code

Figure 4-4. Compact Slave, Process Terminals Tab

The **Process Terminals** tab has the same structure for Compact Slave, Modular Slave, Module, and Signal.

NOTE

Module having subordinate Signal elements can have no process terminals.

When working with the Function Chart Builder:

- In case of an AC 100 Series system, process terminals appear on the right side of the DB element box
- In case of an AC 400 Series system, input process terminals appear on the left side and output process terminals on the right side of the P* type circuit.

Position

The position (order number) of the terminal appears here. It must be a unique integer in the range 0..32767.

In case of an AC 100 Series system, this number describes the order, in which the Function Chart Builder displays the process terminals in the **Edit DB Terminal Values** dialog when you instantiate the DB element.

In case of an AC 400 Series system, the order of type circuit terminals within the read, write, and diag data group corresponds to their relative value in **Position**.

Edit here the position (order number) of existing terminals.
See also (Compact Slave,) Window Names Tab.

To edit a new terminal, scroll down in the list to the last item with position 0, enter a name first in the **Name** column, then change the position.

The list appears ordered by position after each new start of the PROFIBUS Library Editor.

Name

Edit the name of terminals here. It can be 1 to 8 characters long. Follow the syntax of DB terminal names, see Section A.3.2, Allowed Characters for DB Terminal Names.

Data Type Conversion

Set the AMPL data type and additional data type conversion of the process terminal value.

If you create an element on the basis of a *.GSD file, the PROFIBUS Library Editor generates the most plausible AMPL data type and conversion corresponding to the CFG Data of the slave. You can select also other type or conversion based on the documentation of the slave device.

The summary Table 4-2 contains the possible selections.

If you create an element by hand or you change the CFG Data, consult of the documentation of the slave device and see the dedicated tables of Section A.8, Possible Conversions of DP Data Types in case of DP data types and those of Section A.9, Evaluation of DPV1 Data Type Bytes in case of DPV1 data types.

For more information see Section A.12, Background of Data Type Conversion in AC 100 Series.

Table 4-2. Data Type and Conversion (Summary)

Setting	AMPL Data Type	PROFIBUS-DP Data Type	PROFIBUS-DPV1 Data Type	Bit Numbers	Offset Step
B	B	Byte, Word	Boolean	0	1
B<->Byte Packed	B, 8 signals	Byte(1)	Octet String(1)	0, 1,…7	1
B<->Word Packed	B, 16 signals	Word(2)	-	0, 1,…15	1
I	I	Byte, Word	Integer8, Unsigned8, Visible String, Octet String; Integer16, Unsigned16(3)	0	1
I<->DP 2 Bytes	I	2 Bytes	-	0	2
IL	IL	Byte, Word	Unsigned16, Integer32, Unsigned32(4)	0	1
R	R	Byte, Word(5)	Floating Point	*n*(6), 0	1

(1) The 8 bits of this byte are supposed to carry boolean information.
(2) The 16 bits of this word are supposed to carry boolean information.
(3) Consider that the value set of Unsigned16 is 0..65535 and the value set of I(nteger) is -32768..32767.
(4) Consider that the value set of Unsigned32 is 0..(2^32-1) and the value set of I(nteger)L(ong) is -(2^31)..+(2^31-1).
(5) Conversion to R(eal) for PROFIBUS-DP slaves is supported only by newer AC 100 Series node types such as AC 160 2.0.
(6) In case of conversion to R(eal) for PROFIBUS-DP slaves by AC 100 Series targets the bit number *n* defines the analog value resolution on the PROFIBUS-DP side. The analog value starts always at bit 0 (least significant bit) of the concatenated bytes or words and ends at bit *n*-1 (most significant bit). All bits set to 0 correspond to 0%, all bits set to 1 correspond to 100%.

Data Group

Set the data group (direction) of the process terminal. It corresponds to the PC connection data type of the terminal in the following way:

Table 4-3. Data Group

Data Group	PC Connection Data Type	Data Contents
read	r (readable by PC)	Process data
write	r/w (read- and writable by PC)	Process data
diag	r (readable by PC)	Diagnostic data

NOTES

Compact Slave element types can have process terminals belonging to any data group.

Modular Slave elements can have process terminals belonging only to the diag data group.

Module elements not having subordinate Signal elements can have process terminals belonging only to the read and write data group.

Module elements having subordinate Signal elements can have no process terminals.

Signal elements can have process terminals belonging only to the read and write data group.

Bit Number

In case of B<->Byte Packed and B<->Word Packed conversions the 8 bits of Byte and the 16 bits of Word carry 8 and 16 boolean information, respectively.
Specify here the bit number for every packed binary value in the range 0..7 and 0..15, respectively.

In case of conversion to R(eal) for PROFIBUS-DP slaves by AC 100 Series targets the concatenated bytes or words carry an analog value starting at bit 0 (least significant bit) of the first byte or word and ending at some bit n-1 (most significant bit). All bits set to 0 correspond to 0%, all bits set to 1 correspond to 100%.
So, define by bit number n the analog value resolution on the PROFIBUS-DP side.

The bit number is 0 (not used) in all other cases.

NOTE

In case of process terminals of Signal element types set the bit number to 0.
At the instantiation of the parent Module DB element, the Function Chart Builder automatically creates instances of the Signal DB element pertaining to bit numbers 0, 1,…7 or 0, 1, …15, depending on the data type conversion.

Offset

The offset is the sequence number of buffering the PROFIBUS data in the PROFIBUS-DP interface of processor module PM634. It is related also to the position of data in the PROFIBUS telegrams (Protocol Data Units) during data transmission over the PROFIBUS: the first data in telegram is buffered first, then the second data, and so on. Consult the *.GSD file and the documentation of the PROFIBUS device about data position in telegrams.

The bytes of PROFIBUS telegrams are numbered in the range 0..243. In contrast to that, the offset is the order number of data buffered at once, where “data” may occupy 1, 2, or 4 bytes depending on its data type.

Accordingly, you assign the offset numbers 0, 1, and so on, to the first, second, and so on, "data", respectively. Corresponding to data telegrams of different contents, you do so separately in the read data group, write data group, and diag data group.

The offset range varies between 0..243 and 0..60 depending on the occurrence of data with 1, 2, or 4 bytes length in the given group.

The offset step is normally 1.

In case of I<->DP 2 Bytes conversion, the offset step is 2 because in this special case the I(nteger) corresponds to two Bytes building two consecutive data.to define the offset.

NOTES

In case of Module and Signal element types set the offset of the first read and write process terminals normally to 0 and the offsets of the further process terminals relative to these first terminals.

The Module and Signal elements get their absolute offsets at their instantiation:
In the Function Chart Builder, assign appropriate values to the POSITION values in the Base part of the Module instances. So you ensure that the order of Modules corresponds to the specification in the *.GSD file, in fact, to the order of the pertaining data in the PROFIBUS telegrams.
Remember, at the instantiation of a Module, the Function Chart Builder creates automatically also the pertaining Signal instances.

Duplicate Terminal

Activate this button to duplicate the selected terminal and insert it at the last position.

Delete Terminal

Activate this button to delete the selected terminal.

Max. *n* DIAG Terminals in AC 400 Code

This control is available only in *AC 400 and AC 100* preferred mode.

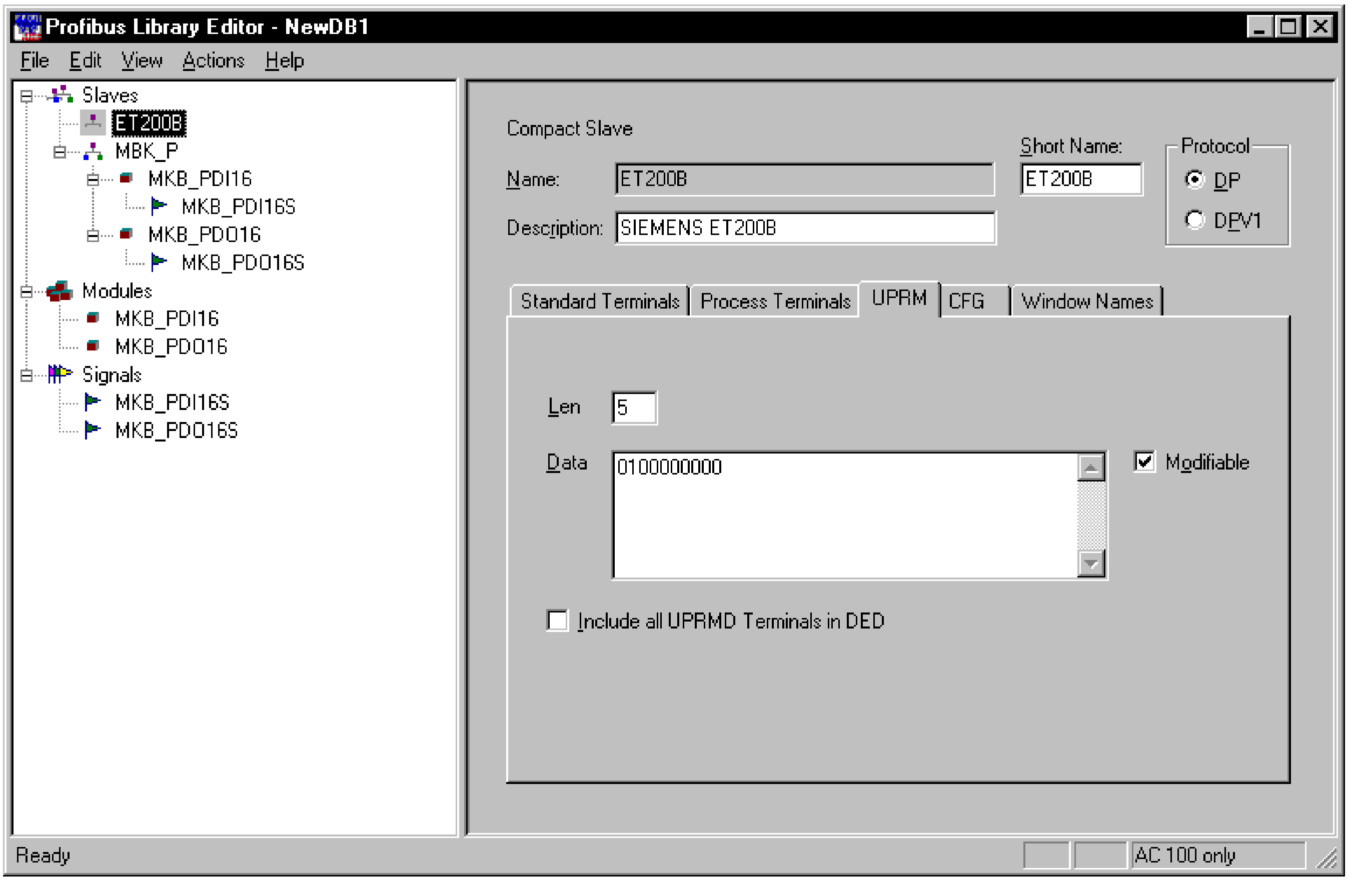
Edit the number of DIAG terminals that have to appear in the P* type circuit source code. That is the way you can eliminate unnecessary DIAG terminals of P* type circuits.

You can set also **all** to let all DIAG terminals appear.

NOTE

Unnecessary DIAG terminals may inflate P* type circuits so that they do not fit into a function chart page, no matter which template and page layout are selected.

Compact Slave, UPRM Tab



Compact Slave

Name: ET200B

Short Name: ET200B

Description: SIEMENS ET200B

Protocol: ● DP ○ DPV1

Len 5

Data 0100000000

☑ Modifiable

☐ Include all UPRMD Terminals in DED

Figure 4-5. Compact Slave, UPRM Tab

The **UPRM** tab has the same structure for all PROFIBUS element types.

The UPRM data are part of the GSD file.

You can also edit them.

Len

If you enter default values in **Data**, the PROFIBUS Library Editor shows in the corresponding **Len** field the data length in bytes.

If you increase the length directly here, 00 values will be added automatically to the corresponding **Data** field.

If you decrease the length directly here, the corresponding **Data** field will be truncated.

Data

Edit here the default values for user parameter data.

Define byte values by entering hexadecimals 0, 1, …, F.
Length can be 0 to 244 bytes.

Modifiable

Select this check box to allow modification of UPRMD values of the corresponding DB element instances in the Function Chart Builder.

Unselect this check box to fix the UPRMD values for the corresponding DB element instances.

Include all UPRMD Terminals

Select this check box to include the maximal number of UPRMD terminals in the DB element type (25 terminals for 244 bytes).

For the first terminals the values entered in **Data** are the default values in the Function Chart Builder, the rest of the terminals have no default values.

Unselect this check box if all default values for UPRM data are specified in **Data**.

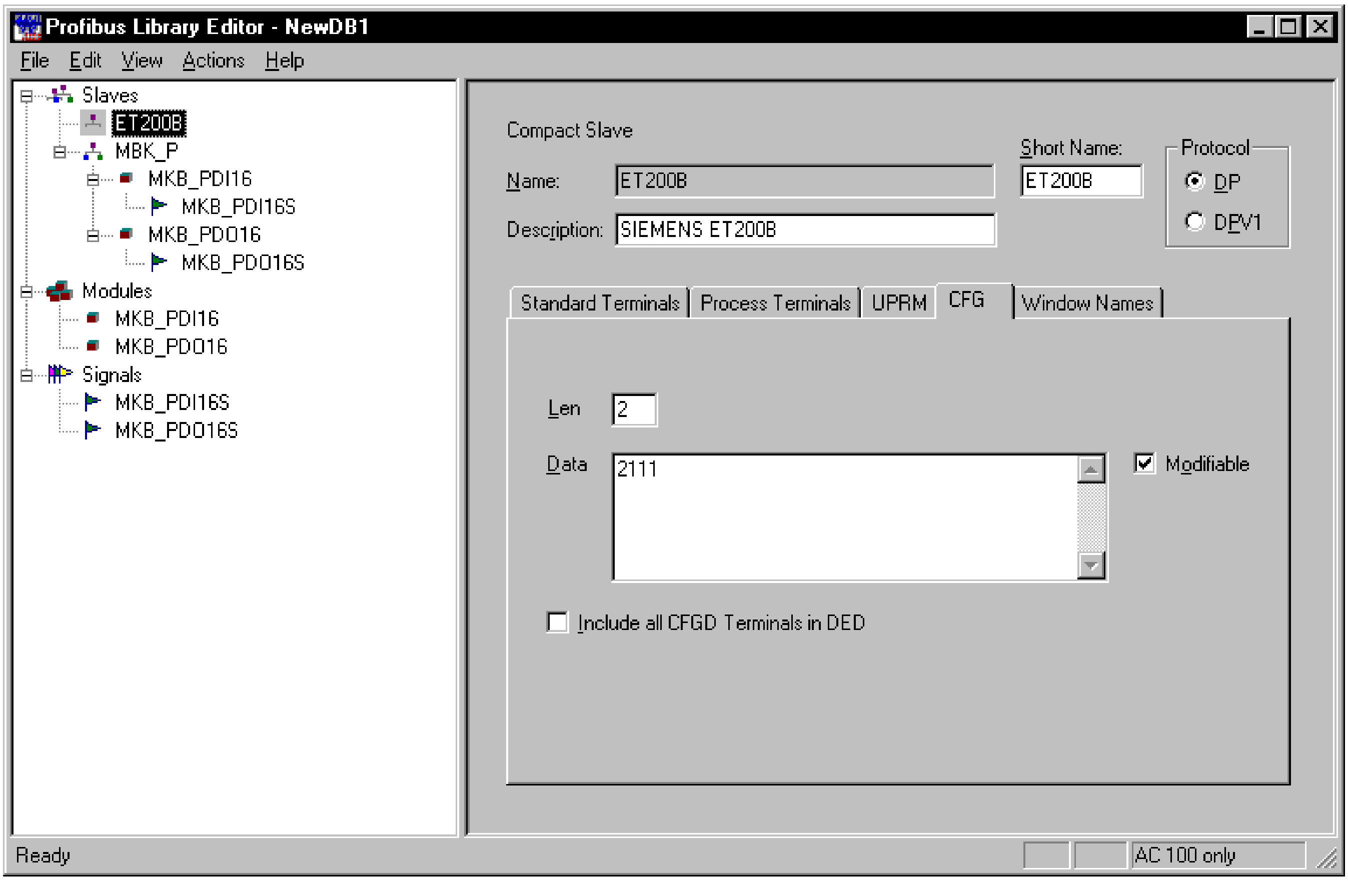
The number of terminals in the DB element for UPRM data are calculated according to the following formula:

number_of_terminals = Len DIV 10 + 1

For example, if Len=0, no terminal including Data is available.

Or, if Len=12, 2 terminals are generated, the first one with the first 10 bytes of **Data**, the second one with the remaining 2 bytes of **Data**.

Compact Slave, CFG Tab



Name: ET200B

Short Name: ET200B

Description: SIEMENS ET200B

Protocol: ● DP ○ DPV1

Len: 2

Data: 2111

☑ Modifiable

☐ Include all CFGD Terminals in DED

Figure 4-6. Compact Slave, CFG Tab

The **CFG** tab has the same structure for all PROFIBUS element types.

The configuration data are part of the GSD file.

You can also edit them.

Len

If you enter default values in **Data**, the PROFIBUS Library Editor shows in the corresponding **Len** field the data length in bytes.

If you increase the length directly here, 00 values will be added automatically to the corresponding **Data** field.

If you decrease the length directly here, the corresponding **Data** field will be truncated.

Data

Edit here the default values for configuration data.

Define byte values by entering hexadecimals 0, 1, …, F.

Length can be 0 to 244 bytes.

Modifiable

Select this check box to allow modification of CFGD values of the corresponding DB element instances in the Function Chart Builder.

Unselect this check box to fix the CFGD values for the corresponding DB element instances.

Include all CFGD Terminals in DED

Select this check box to include the maximal number of CFGD terminals in the DB element (25 terminals for 244 bytes).

For the first terminals the values entered in **Data** are the default values in the Function Chart Builder, the rest of the terminals have no default values.

Unselect this check box if all default values for CFG data are specified in **Data**.

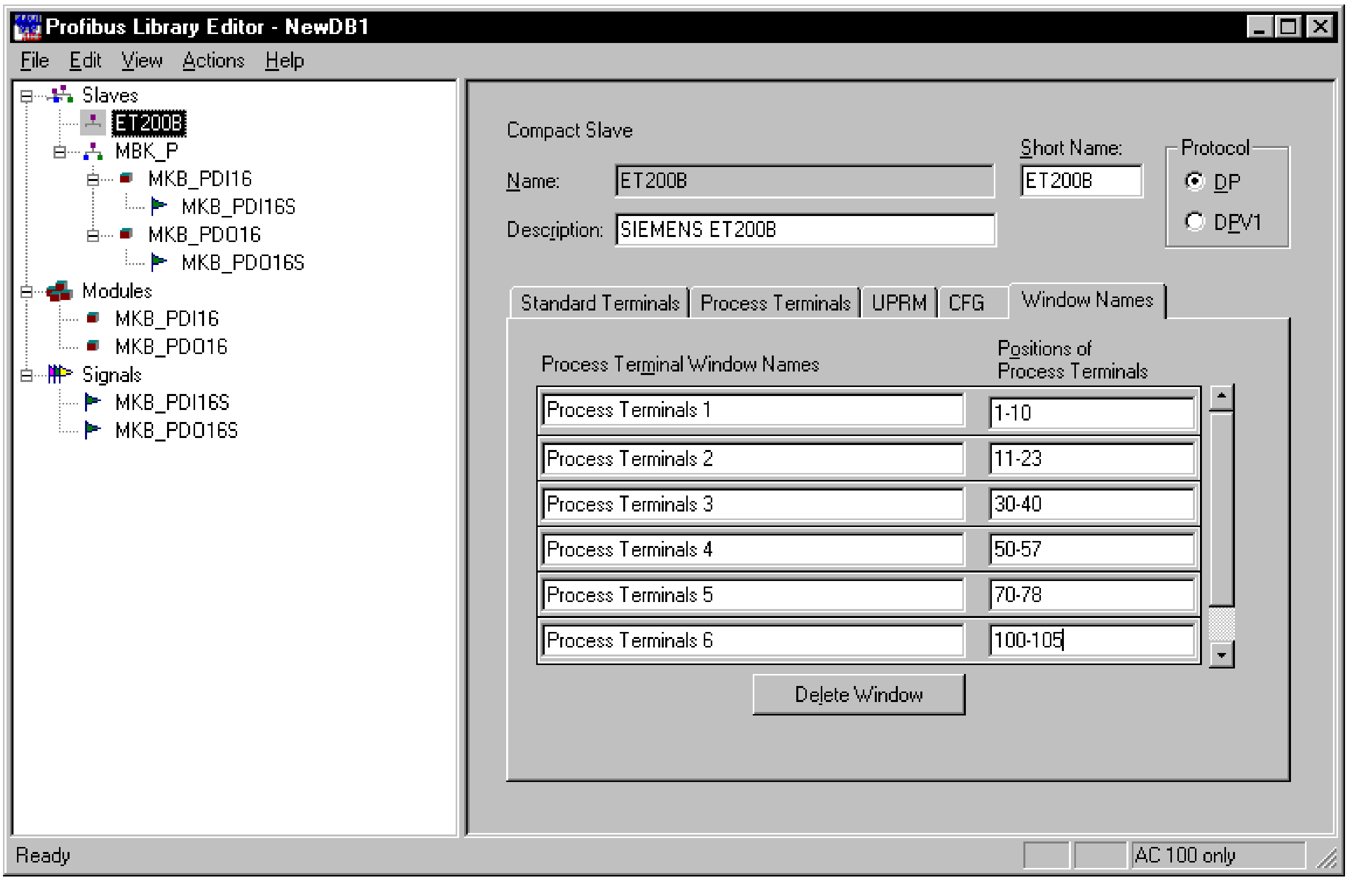
The number of terminals in the DB element for CFG data are calculated according to the following formula:

number_of_terminals = Len DIV 10 + 1

For example, if Len=0, no terminal including Data is available.

Or, if Len=12, 2 terminals are generated, the first one with the first 10 bytes of **Data**, the second one with the remaining 2 bytes of **Data**.

Compact Slave, Window Names Tab



Compact Slave

Name: ET200B

Short Name: ET200B

Description: SIEMENS ET200B

Protocol
● DP
○ DPV1

Process Terminal Window Names	Positions of Process Terminals
Process Terminals 1	1-10
Process Terminals 2	11-23
Process Terminals 3	30-40
Process Terminals 4	50-57
Process Terminals 5	70-78
Process Terminals 6	100-105

AC 100 only

Figure 4-7. Compact Slave, Window Names Tab

The **Window Names** tab has the same structure for all PROFIBUS element types. It has no significance for Advant Controller 400 Series when you are working in *AC 400 and AC 100* preferred mode.

The **Window Names** tab includes up to 8 lines.

As displayed in the **Edit DB Terminal Values** dialog in the Function Chart Builder, a Slave, Module, and Signal DB element can have 1 to 11 windows (DB element parts, graphic symbols).

NOTES

Anyway, the PROFIBUS Library Editor creates automatically a number of windows: one Base part, one or two User_Prm_Data and Gfg_Data depending on length, and for Slaves also Slave parameters.

Even if you do not specify window names in this tab for process terminals, the PROFIBUS Library Editor creates automatically an appropriate number of windows with standard names Process Values, Process Values 1, and so on.

In this tab, you can specify window names for process terminals additionally or instead of the standard ones. If there are some process terminals not covered by any of the ranges in the **Positions of Process Terminals** column, the PROFIBUS Library Editor creates an appropriate number windows with standard names Process Values, Process Values 1, and so on, in which these left out terminals can be displayed.

A message is displayed during execution of the **Action | Generate DED for AC 100** command if the maximal number 11 is exceeded and specified DB element windows cannot be included.It contains up to 8 line.

Process Terminal Window Names

Edit here the DB element window names to be used for process terminals in Function Chart Builder.

The default names are `Process Values 1` to `Process Values 8`.

Positions of Process Terminals

Edit here the position of process terminals included in the **Process Terminals** tab to specify the range of process terminals in a window in the following form:

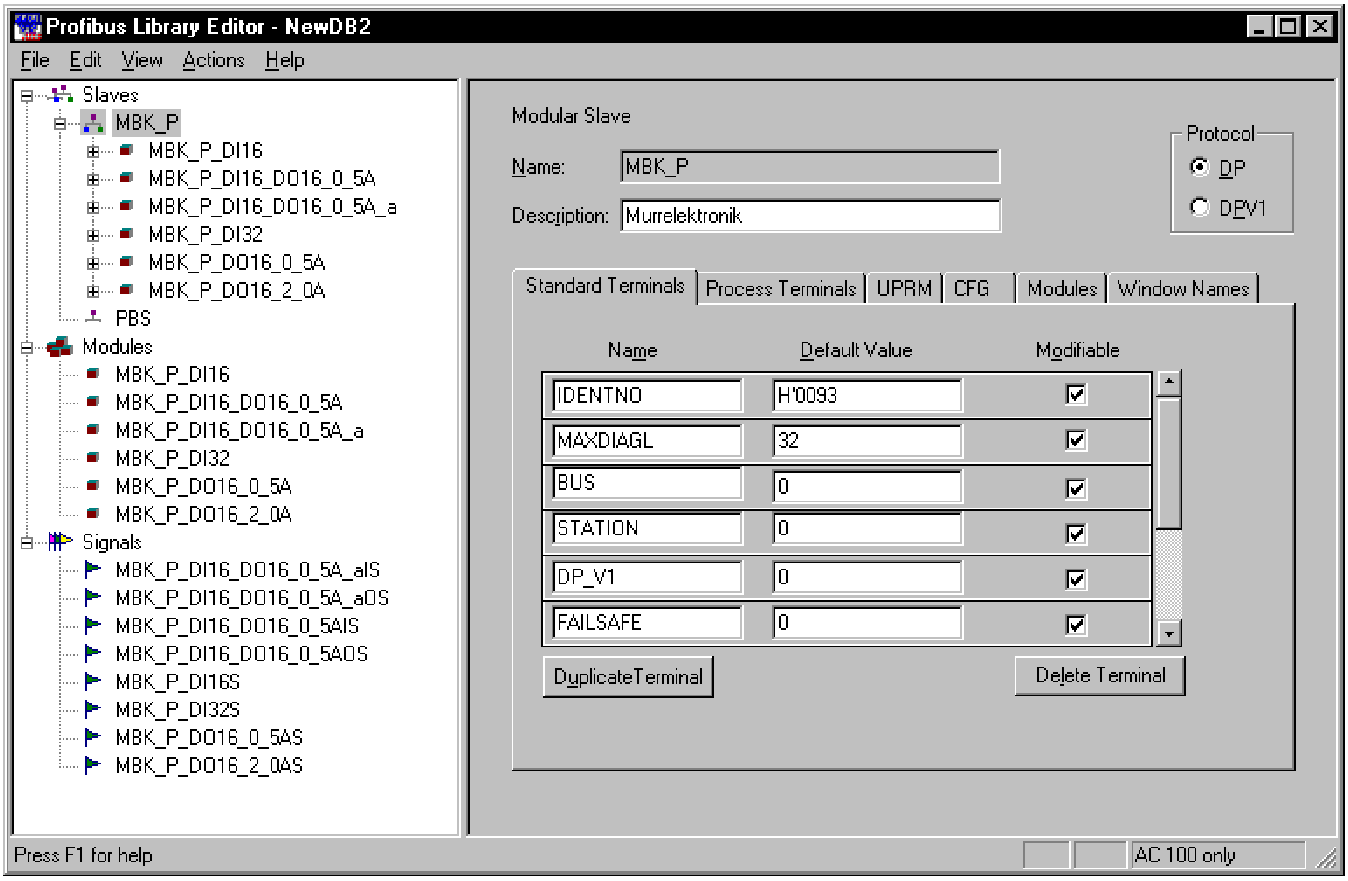
from_number `-´ *to_number*
,where 0 <= *from_number* <= *to_number* <= 32767.

Number of terminals in a window is 0 to 18.
A position can occur in no more than one window.

Delete Window

Activate this button to delete the selected window.

Modular Slave, Standard Terminals Tab



Modular Slave

Name: MBK_P

Description: Murrelektronik

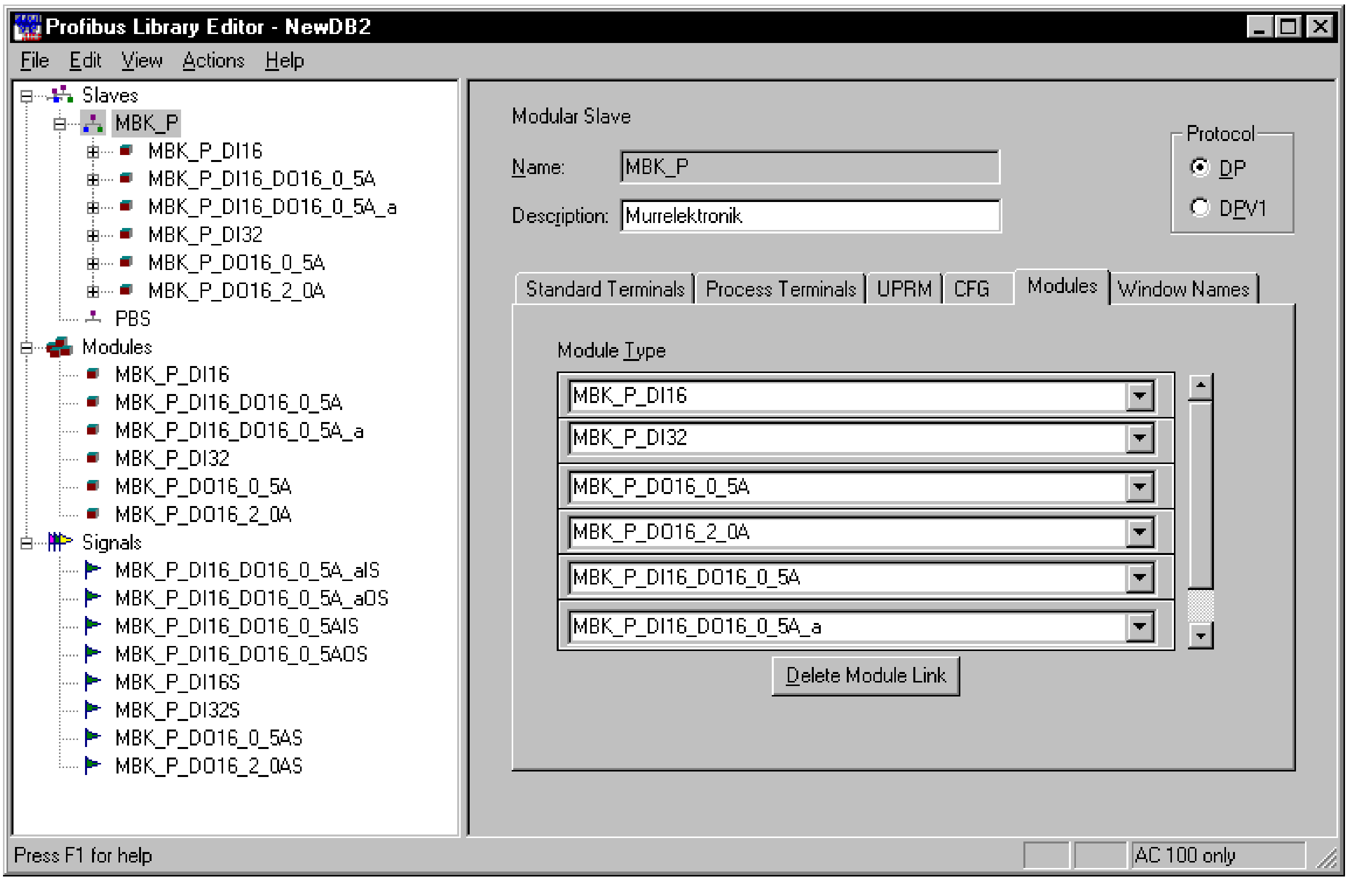
Protocol: ● DP ○ DPV1

Name	Default Value	Modifiable
IDENTNO	H'0093	☑
MAXDIAGL	32	☑
BUS	0	☑
STATION	0	☑
DP_V1	0	☑
FAILSAFE	0	☑

Figure 4-8. Modular Slave, Standard Terminals Tab

For description of controls of the **Standard Terminals** tab, see Compact Slave, Standard Terminals Tab.

Modular Slave, Modules Tab



Modular Slave

Name: MBK_P

Description: Murrelektronik

Protocol
● DP
○ DPV1

Module Type

MBK_P_DI16
MBK_P_DI32
MBK_P_DO16_0_5A
MBK_P_DO16_2_0A
MBK_P_DI16_DO16_0_5A
MBK_P_DI16_DO16_0_5A_a

Figure 4-9. Modular Slave, Modules Tab

The **Modules** tab has the following controls:

Module Type

Enter all module types belonging to the modular station. Select module types from the available ones listed in the combo box. New module types cannot be entered.
The entered module types are also shown in tree view.
The maximal number of module types in a modular slave is 32.

Delete Module Link

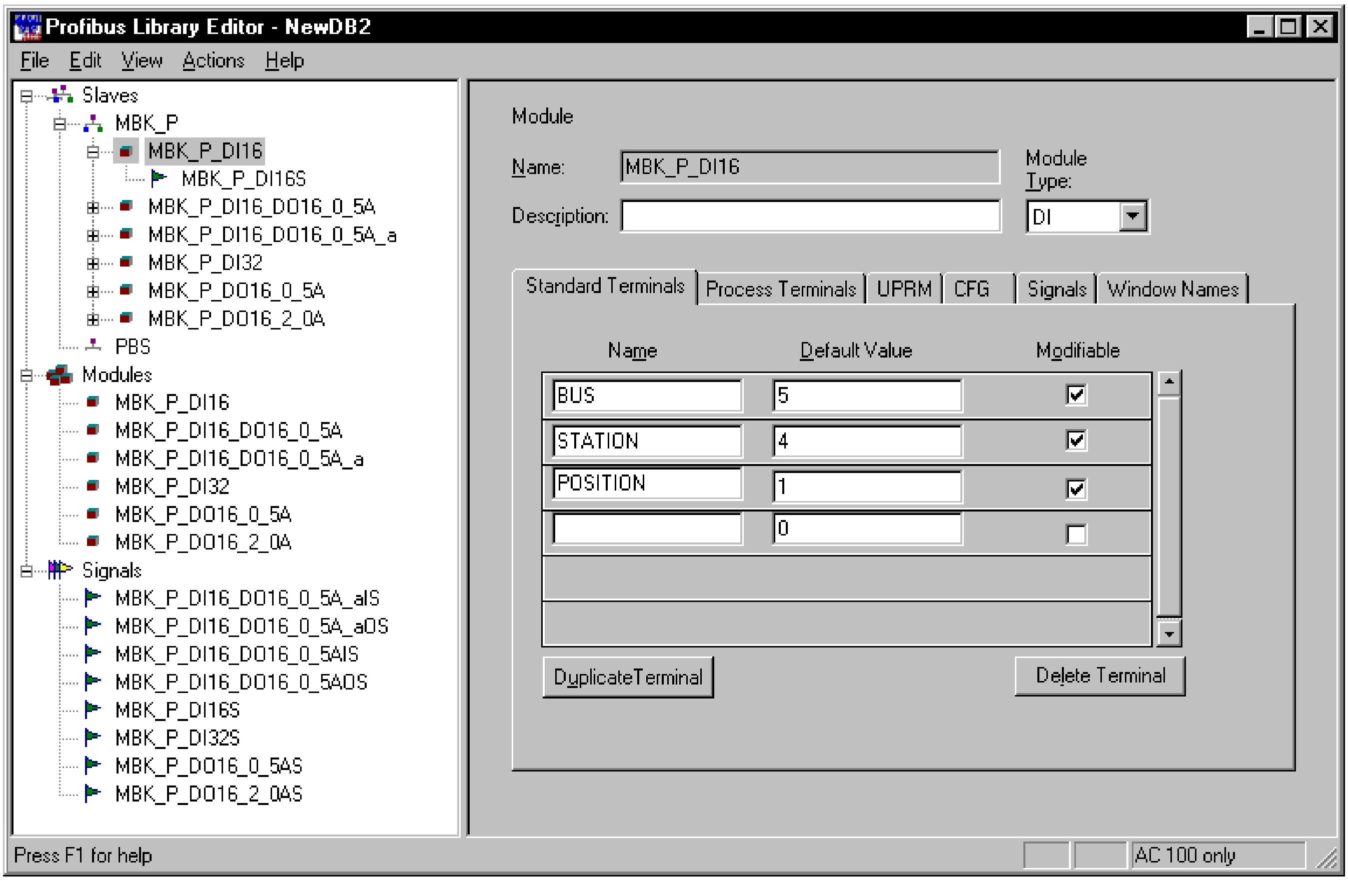
Activate this button to delete the currently selected module type link.

NOTE

A module type can be used in more than one Modular Station.

For description of common tabs, see Compact Slave, Process Terminals Tab, and so forth.

Module, Standard Terminals Tab



Name: MBK_P_DI16

Description:

Module Type: DI

Name	Default Value	Modifiable
BUS	5	☑
STATION	4	☑
POSITION	1	☑
	0	☐

Figure 4-10. Module, Standard Terminals Tab

Common control element of modules in the upper part of the right pane is the following:

Module Type

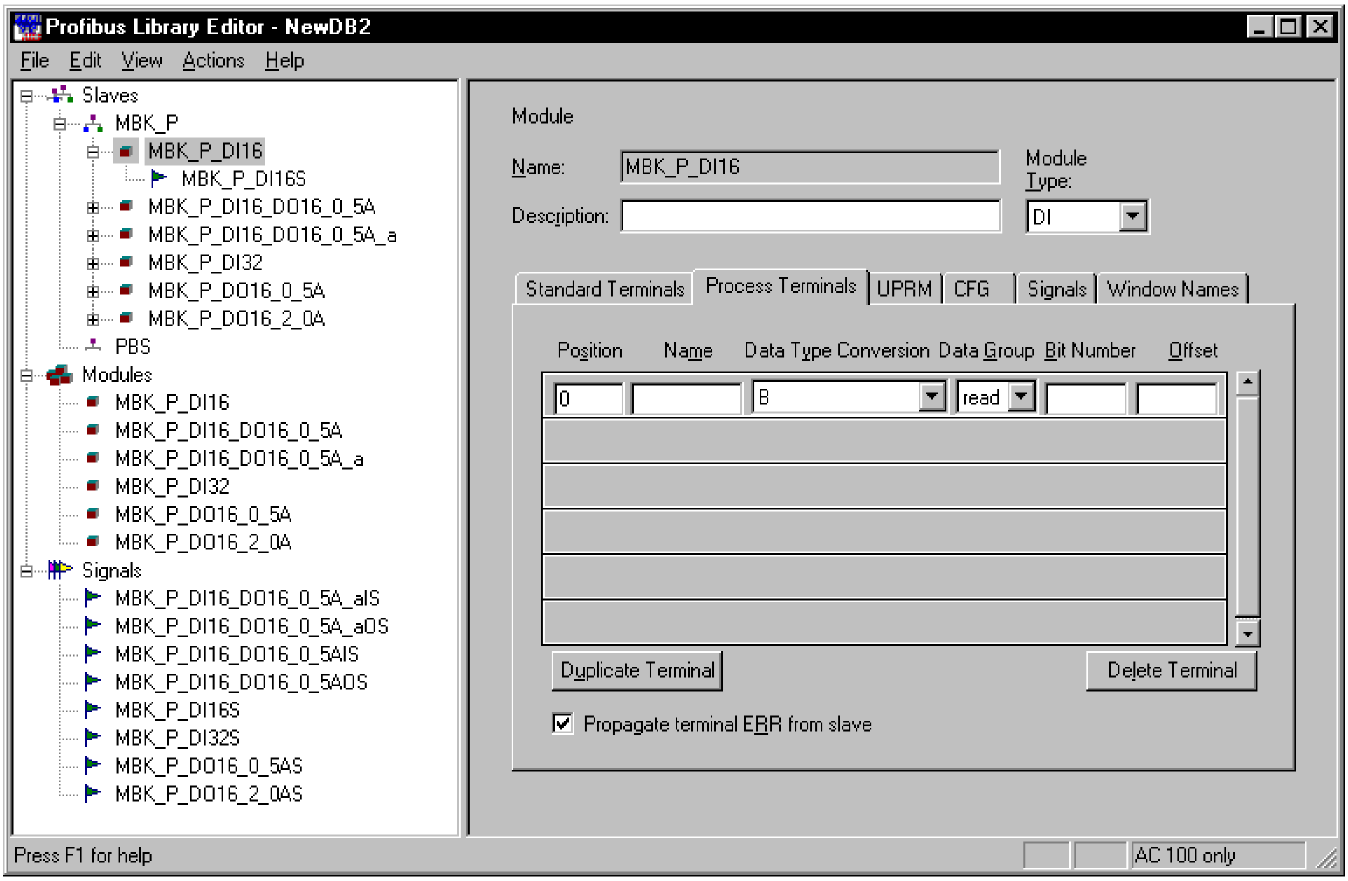
Enter the type of the selected module.

The possible module types, included in the combo box are: AI, AO, DI, DO, AX, DX.

See also Section A.11, Signal Types Depending on Module Type.

For description of controls of the **Standard Terminals** tab, see Compact Slave, Standard Terminals Tab.

Module, Process Terminals Tab



Name: MBK_P_DI16

Description:

Module Type: DI

Position	Name	Data Type Conversion	Data Group	Bit Number	Offset
0		B	read		

☑ Propagate terminal ERR from slave

Figure 4-11. Module, Process Terminals Tab

Special control of the **Process Terminals** tab for modules is the following:

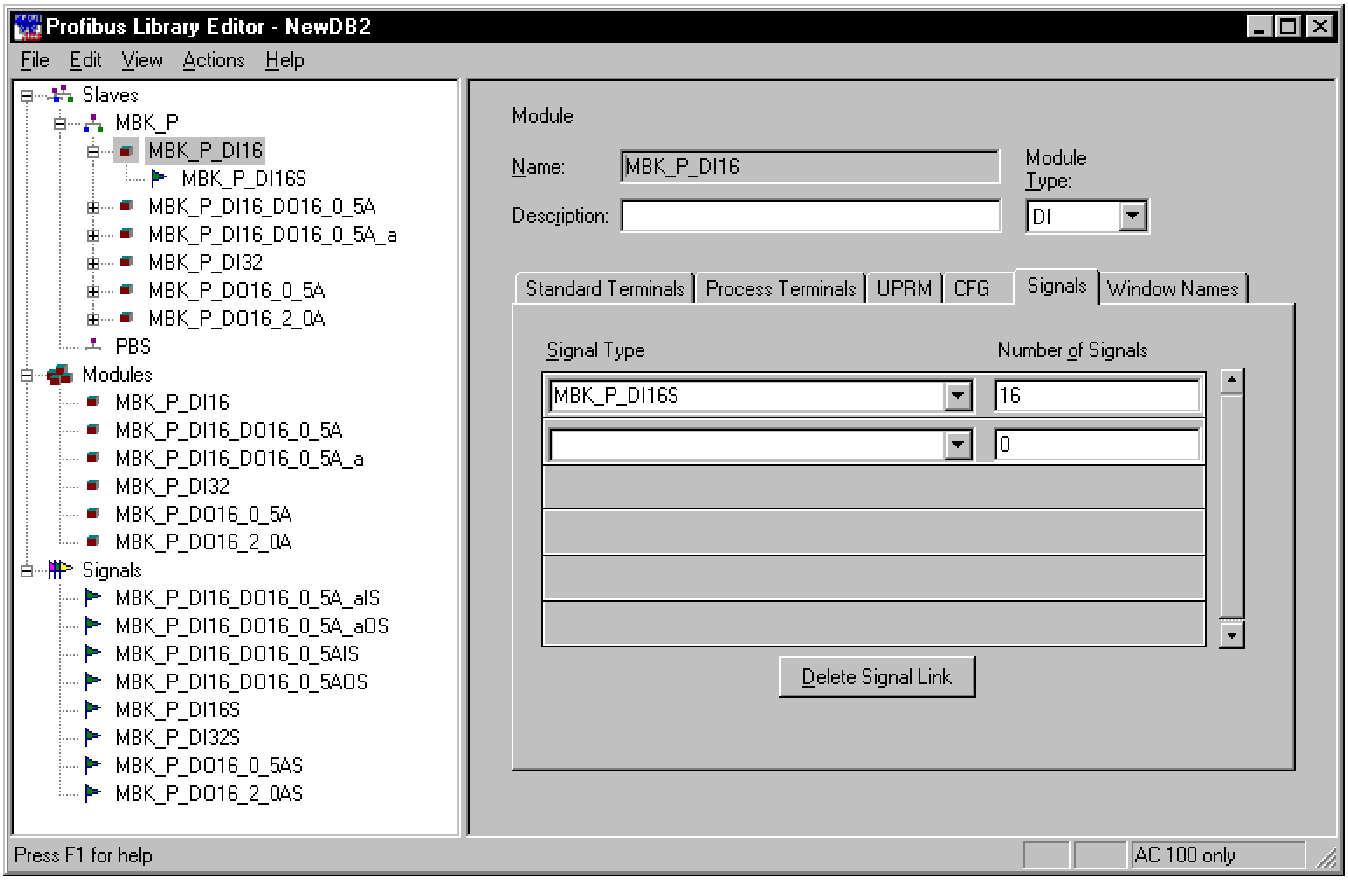
Propagate Terminal ERR from Slave

Select this check box to propagate the value of the ERR terminal of the modular slave to the ERR terminal of the module.

Unselect this check box in case of modules with own error conditions.

For description of the other controls of the tab, see Compact Slave, Process Terminals Tab.

Module, Signals Tab



Name: MBK_P_DI16

Description:

Module Type: DI

Signal Type	Number of Signals
MBK_P_DI16S	16
	0

Figure 4-12. Module, Signals Tab

The **Signals** tab has the following controls:

Signal Type

Enter all signal types belonging to the selected module type.
Select the signal types from the available signal types listed in the combo box.
New signal types cannot be entered.

The entered signal types are also shown in tree view.

Number of Signals

Enter all number of signals.

The number of signals is automatically created in Function Chart Builder if an instance of the module element is created.

Delete Signal Link

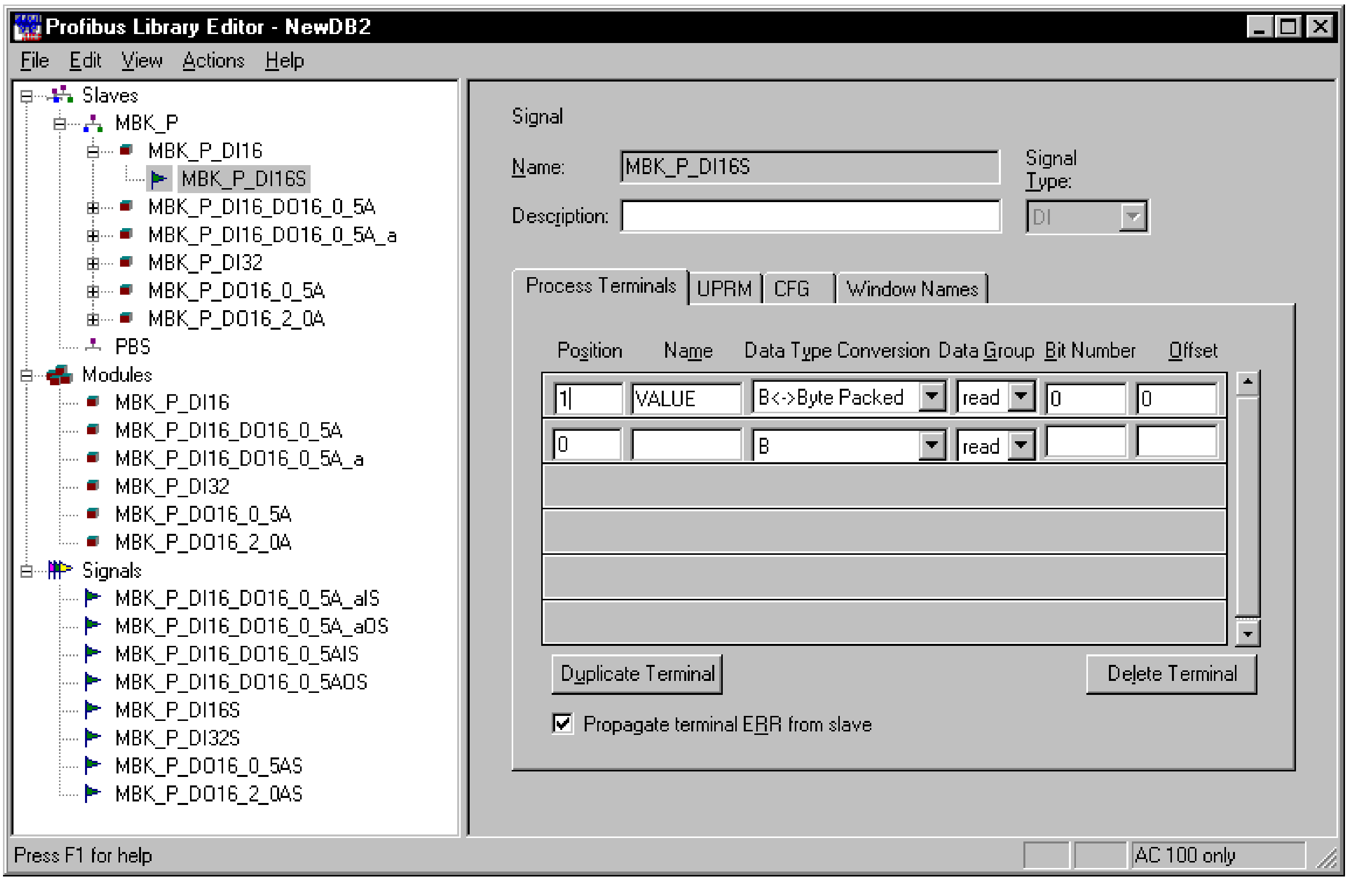
Activate this button to delete the currently selected signal type link.

NOTE

A signal type can be used for only one module type.
Already used signal types are marked in tree view.

For description of common tabs, see Compact Slave, UPRM Tab, and so forth.

Signal, Process Terminals Tab



Name: MBK_P_DI16S

Description:

Signal Type: DI

Position	Name	Data Type Conversion	Data Group	Bit Number	Offset
1	VALUE	B<->Byte Packed	read	0	0
0		B	read		

☑ Propagate terminal ERR from slave

Figure 4-13. Signal, Process Terminals Tab

Common control element of signals in the upper part of the right pane is the following:

Signal Type

Enter the type of the selected signal.
The possible signal types, included in the combo box are: AI, AO, DI, DO.
See also Section A.11, Signal Types Depending on Module Type.

Special control of the **Process Terminals** tab for signals is the following:

Propagate Terminal ERR from Slave

Select this check box to propagate the value of the ERR terminal of the modular slave to the ERR terminal of the signal.
Unselect this check box in case of signals with own error conditions.

For description of the other controls of the tab, see Compact Slave, Process Terminals Tab.

For description of common tabs, see Compact Slave, UPRM Tab, and so forth.

4.4.1.3 Context Menu of Left Pane

If you click with the right mouse button in the left pane on an element, the PROFIBUS Library Editor shows the following context menu:

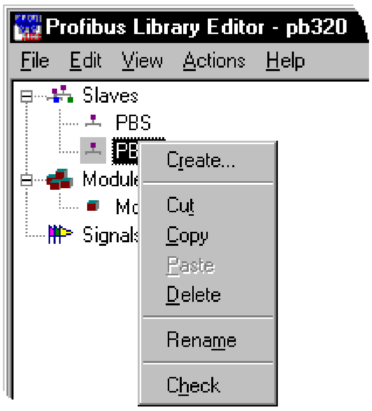


Figure 4-14. Context Menu of Left Pane, Inside

If you click with the right mouse button in the left pane outside the elements, the PROFIBUS Library Editor shows the following context menu:

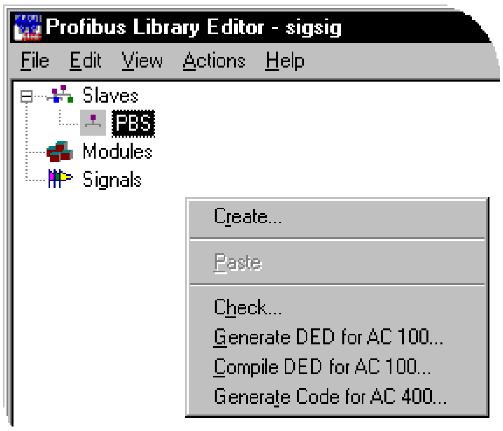

Profibus Library Editor - sigsig
File
Edit
View
Actions
Help
Slaves
PBS
Modules
Signals
Create...
Paste
Check...
Generate DED for AC 100...
Compile DED for AC 100...
Generate Code for AC 400...

Figure 4-15. Context Menu of Left Pane, Outside

4.4.1.4 Context Menu of Right Pane

If you click with the right mouse button in the right pane, the PROFIBUS Library Editor shows the following context menu:

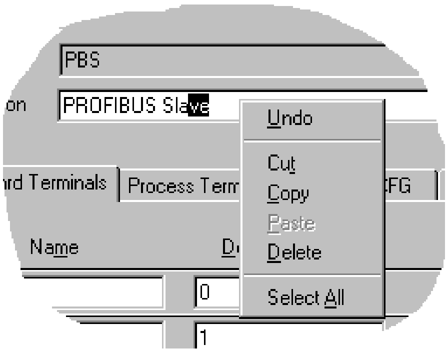


Figure 4-16. Context Menu of Right Pane

4.4.1.5 Preview Window

The PROFIBUS Library Editor displays this window during creation of elements semi-automatically on the basis of a *.GSD file.

In case of Compact Slave descriptions:

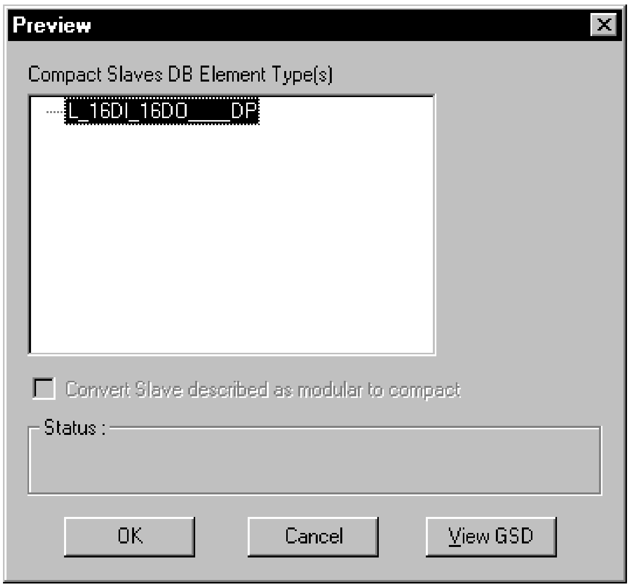
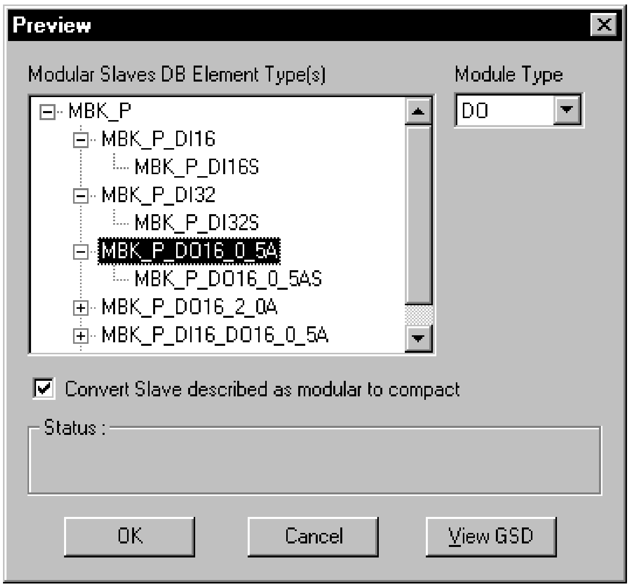

Preview
Compact Slaves DB Element Type(s)
L_16DI_16DO___DP
☐ Convert Slave described as modular to compact
Status :
OK
Cancel
View GSD

Figure 4-17. Preview Window, Compact Slave

In case of Modular Slave descriptions you can see the element structure. If you select a module, you can check and set the module type:



Modular Slaves DB Element Type(s)

- MBK_P
  - MBK_P_DI16
    - MBK_P_DI16S
  - MBK_P_DI32
    - MBK_P_DI32S
  - **MBK_P_DO16_0_5A**
    - MBK_P_DO16_0_5AS
  - MBK_P_DO16_2_0A
  - MBK_P_DI16_DO16_0_5A

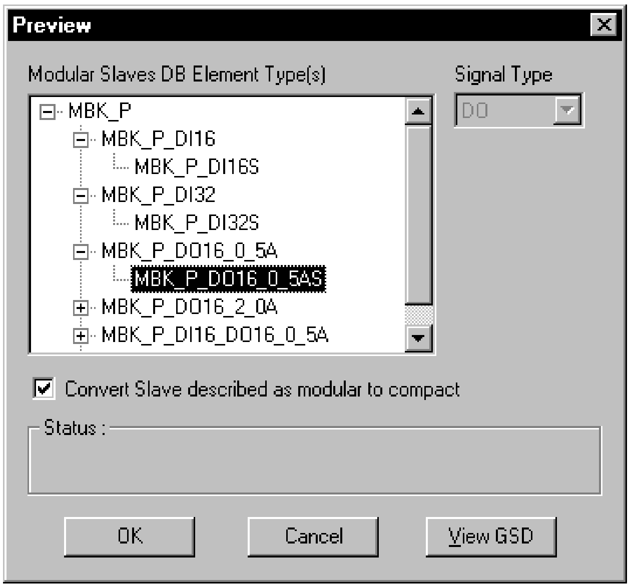
Module Type: DO

☑ Convert Slave described as modular to compact

Status :

Figure 4-18. Preview Window, Modular Slave, Module Selected

If you select a signal, you can check and set the signal type:



Modular Slaves DB Element Type(s)

- MBK_P
  - MBK_P_DI16
    - MBK_P_DI16S
  - MBK_P_DI32
    - MBK_P_DI32S
  - MBK_P_DO16_0_5A
    - **MBK_P_DO16_0_5AS**
  - MBK_P_DO16_2_0A
  - MBK_P_DI16_DO16_0_5A

Signal Type: DO

☑ Convert Slave described as modular to compact

Status :

Figure 4-19. Preview Window, Modular Slave, Signal Selected

Suggested Element Types

This structure tree shows the DB element types to be created by the PROFIBUS Library Editor on the basis of the selected *.GSD file.

Accept them by activating the **OK** button.

If necessary. you can rename them later by the **Edit | Rename** command.

Refuse them by activating the **Cancel** button.

In the latter case, the PROFIBUS Library Editor shows again the **Open GSD File** dialog allowing you to select another *.GSD file.

Module Type

Check and select here the module type of the selected module DB element.

Select

- AI for modules carrying analog inputs
- AO for modules carrying analog outputs
- AX for modules carrying either analog inputs or outputs, possibly also digital inputs and/or outputs
- DI for modules carrying digital inputs
- DO for modules carrying digital outputs.

See also Section A.11, Signal Types Depending on Module Type.

For description of **Suggested Element Types** and **View GSD**, see before.

Signal Type

Check and select here the signal type of the selected signal DB element.

Select

- AI for analog input signals
- AO for analog output signals
- DI for digital input signals
- DO for digital output signals.

Convert Slave Described as Modular to Compact

Select this check box if you want to convert the selected slave described as modular.
Then it will be represented as a compact slave.

Unselect this check box if you want to represent modular slaves that are described as modular in the *.GSD file.

The check box is always set and dimmed if the preferred mode is *AC 400 and AC 100*.

It is not available for slaves described as compact.

Status

Here you can follow the creation of elements on the basis of the selected *.GSD file.

View GSD

Activate this button to open the selected *.GSD file with the editor associated with this file type.

NOTE

You can associate a file type in Windows NT with an editor by double-clicking on such a file in the Windows NT Explorer.

4.4.2 Commands of PROFIBUS Library Editor

4.4.2.1 File Menu

File | New …

Shortcut: CTRL+N

Use this command to create a new data base.
At the same time, you set the preferred mode for the new database:

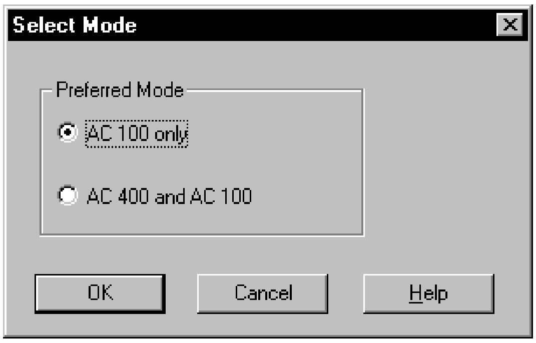


Figure 4-20. Select Mode

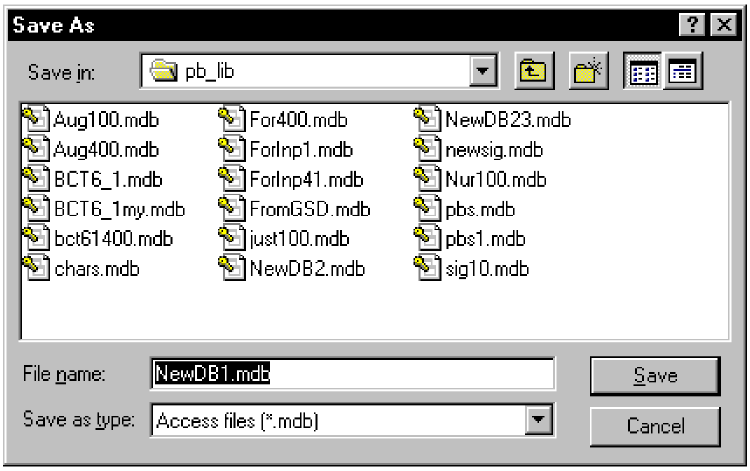
Preferred Mode

Set **AC 100 only** if you intend to project only Advant Controller 100 Series applications containing PROFIBUS devices.

Set **AC 400 and AC 100** if you intend to project Advant Controller 400 Series and Advant Controller 100 Series applications containing PROFIBUS devices.

The PROFIBUS Library Editor copies the PROFIBUS.MDB model data base from `drive:\AdvaBNT\Fcb\release_x\BIN32\db_lib\source` to the `C:\PB_LIB` folder, assigns the name `NewDBnn` to this data base, where *nn* is 1, 2, ….

Accept it or set an appropriate name in the **Save As** dialog:



File name: NewDB1.mdb

Save as type: Access files (*.mdb)

Figure 4-21. Set the Name of the New Data Base

For explanations of the controls in the **Save As** dialog box, see the **File | Save As** command.

The PROFIBUS Library Editor shows the structure of the new data base expanded in the left pane of the application window.

NOTES

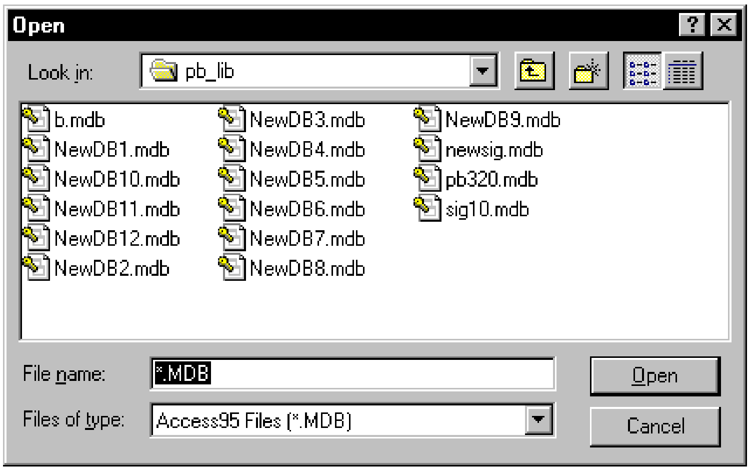
The C:\PB_LIB directory can be changed by setting the environment variable PB$LIB. See Section 4.3.1, Preparation, Check of Installation and Files.

The new data base contains already a PBS compact slave. You can rename or delete it. Take it for a template. It contains all the terminals that appear anyway later in the generated DB element description and ought to have default values.

File | Open …

Shortcut: CTRL+O

Use this command to open an existing data base, that is, an existing *.MDB file.



File name: *.MDB

Files of type: Access95 Files (*.MDB)

Figure 4-22. Open

Look in (Directory)

The symbol shows the directory currently browsed for files.
Default is the C:\PB_LIB directory.

You can also select a subdirectory in the list box below.
Or go to the directory on the next higher level by clicking on the **Up One Level** icon.
Click on the **Create New Folder** icon to create a new subdirectory.

Up One Level and Create New Folder (Icons)

Click on the **Up One Level** icon to go to the directory on the next higher level.
Click on the **Create New Folder** icon to create a new subdirectory.

List and Details (Icons)

Click on these icons to display the files and subdirectories in corresponding forms.

Files and Subdirectories

The list box contains all available files and subdirectories in the specified directory, which correspond to the filter condition.

Click on the **List** and **Details** icons to display the files and subdirectories in corresponding forms.

File Name

Select a file name from the list box or type it here.
Or edit here a new filter condition, for example, MY*.MDB, and press ENTER to make it effective.

Save as Type

Select a filter condition from the standard filters in this drop-down list box.
*.MDB is set as default extension of Access data base files.

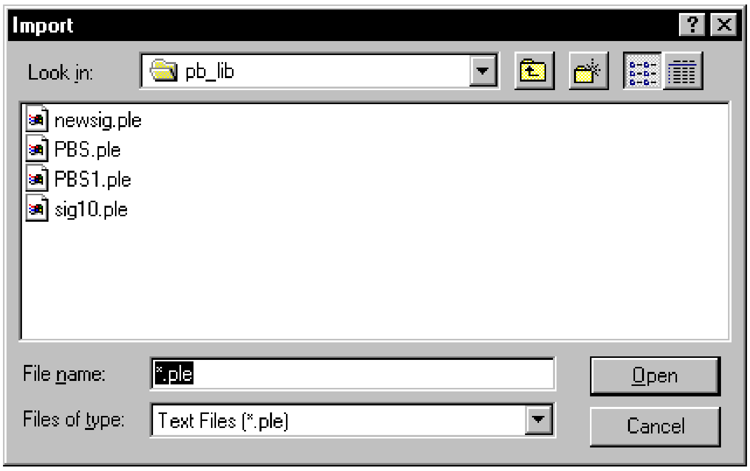
Open

Activate this button to start opening the selected file.

The PROFIBUS Library Editor closes the current (previous) data base, opens the specified one, and shows its structure expanded in the left pane of the application window.

File | Import …

Use this command to import a PROFIBUS library file (an export file) containing one or several element types into the current data base.



Import

Look in: pb_lib

newsig.ple
PBS.ple
PBS1.ple
sig10.ple

File name: *.ple

Files of type: Text Files (*.ple)

Figure 4-23. Import

Look in (Directory)

The symbol shows the directory currently browsed for files.
Default is the C:\PB_LIB directory.

You can also select a subdirectory in the list box below.
Or go to the directory on the next higher level by clicking on the **Up One Level** icon.
Click on the **Create New Folder** icon to create a new subdirectory.

Up One Level and Create New Folder (Icons)

Click on the **Up One Level** icon to go to the directory on the next higher level.
Click on the **Create New Folder** icon to create a new subdirectory.

List and Details (Icons)

Click on these icons to display the files and subdirectories in corresponding forms.

Files and Subdirectories

The list box contains all available files and subdirectories in the specified directory, which correspond to the filter condition.
Click on the **List** and **Details** icons to display the files and subdirectories in corresponding forms.

File Name

Select a file name from the list box or type it here.

Or edit here a new filter condition, for example, MY*.PLE, and press ENTER to make it effective.

Save as Type

Select a filter condition from the standard filters in this drop-down list box.

*.PLE is set as default extension of PROFIBUS library files (export files).

Open

Activate this button to start importing the selected file.

The PROFIBUS Library Editor shows the resulting structure expanded in the left pane of the application window.

File | Save

Shortcut: CTRL+S

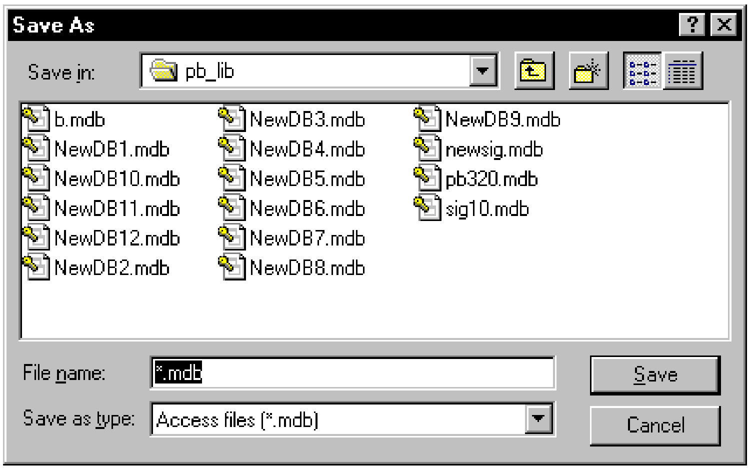
Use this command to save all the modifications that you made in the current data base.

NOTE

If you edit a data base, all changes in a field are effective at once on moving the focus to any other field. With this command you save also the last change without moving the focus to any other field.

File | Save As …

Use this command to save the current data base in another file.



Save in: pb_lib

File name: *.mdb

Save as type: Access files (*.mdb)

Figure 4-24. Save As

Save in (Directory)

The symbol shows the directory currently browsed for files.

Default is the C:\PB_LIB directory.
You can also select a subdirectory in the list box below.
Or go to the directory on the next higher level by clicking on the **Up One Leve**l icon.

Click on the **Create New Folder** icon to create a new subdirectory.

Up One Level and Create New Folder (Icons)

Click on the **Up One Level** icon to go to the directory on the next higher level.
Click on the **Create New Folder** icon to create a new subdirectory.

List and Details (Icons)

Click on these icons to display the files and subdirectories in corresponding forms.

Files and Subdirectories

The list box contains all available files and subdirectories in the specified directory, which correspond to the filter condition.

Click on the **List** and **Details** icons to display the files and subdirectories in corresponding forms.

File Name

Select a file name from the list box or type it here.

Or edit here a new filter condition, for example, MY*.MDB, and press ENTER to make it effective.

Save as Type

Select a filter condition from the standard filters in this drop-down list box.

*.MDB is set as default extension of Access data base files.

Save

Activate this button to start saving to the file with the given name.

The PROFIBUS Library Editor copies the current data base with all the changes if there were any, to the specified new file, and opens the latter. The former data base remains unchanged.

File | Skip Modifications

Use this command to skip modifications, which you made during the current PROFIBUS Library Editor session since the last **File | Save** command in the current data base.

The PROFIBUS Library Editor asks you to confirm skipping.

CAUTION

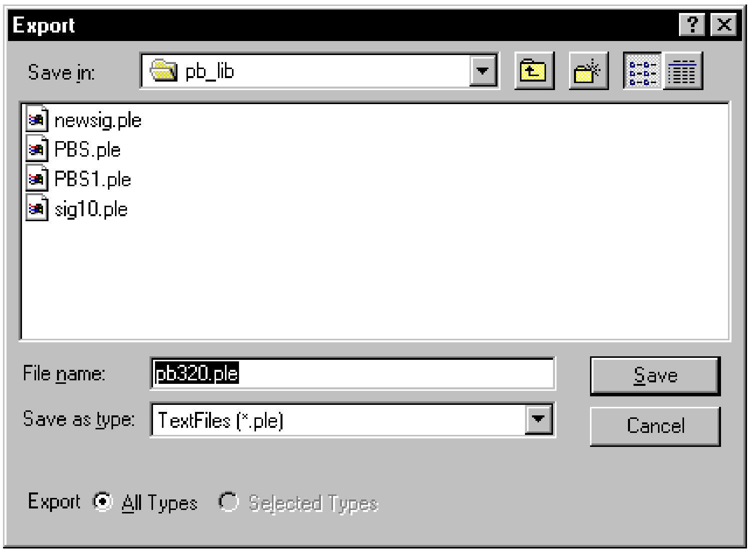
If you answer Yes, you lose all the changes that you made since the last **File | Save**, **File | Save As**, or **File | Exit** command. You cannot undo this command.

File | Compact Database

Use this command to compact the current data base.

File | Export …

Use this command to export either all of the element types of the current data base or only the selected one and—if there are any—its subordinates.



Save in: pb_lib

newsig.ple
PBS.ple
PBS1.ple
sig10.ple

File name: pb320.ple

Save as type: TextFiles (*.ple)

Export ● All Types ○ Selected Types

Figure 4-25. Export

Save in (Directory)

The symbol shows the directory currently browsed for files.

Default is the C:\PB_LIB directory.
You can also select a subdirectory in the list box below.
Or go to the directory on the next higher level by clicking on the **Up One Level** icon.

Click on the **Create New Folder** icon to create a new subdirectory.

Up One Level and Create New Folder (Icons)

Click on the **Up One Level** icon to go to the directory on the next higher level.
Click on the **Create New Folder** icon to create a new subdirectory.

List and Details (Icons)

Click on these icons to display the files and subdirectories in corresponding forms.

Files and Subdirectories

The list box contains all available files and subdirectories in the specified directory, which correspond to the filter condition.

Click on the **List** and **Details** icons to display the files and subdirectories in corresponding forms.

File Name

Select a file name from the list box or type it here.

Or edit here a new filter condition, for example, MY*.PLE, and press ENTER to make it effective.

Save as Type

Select a filter condition from the standard filters in this drop-down list box.

*.PLE is set as default extension of PROFIBUS library files (export files).

Save

Activate this button to start saving to the file with the given name.

Export

Select the **All Types** option to export all of the element types of the current data base.

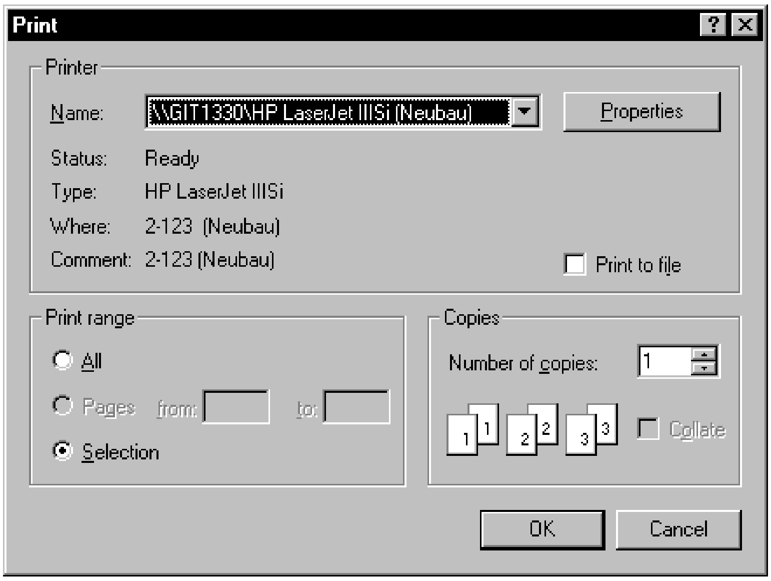
Select the **Selected Types** option to export only the selected one and—if there are any—its subordinates.

The PROFIBUS Library Editor saves the element type(s) in the PROFIBUS library file (export file).

File | Print …

Shortcut: CTRL+P

Use this command to print either all of the element types of the current data base or only the selected one and—if there are any—its subordinates.



Name: \\GIT1330\HP LaserJet IIISi (Neubau)
Status: Ready
Type: HP LaserJet IIISi
Where: 2-123 (Neubau)
Comment: 2-123 (Neubau)
☐ Print to file

○ All
○ Pages from: to:
● Selection

Number of copies: 1
☐ Collate

Figure 4-26. Print

Printer

Select one of the current installed printers shown in the drop-down list box you want to use. The static text shows status, type, and location of the selected printer.

You install printers and configure ports using the **Settings | Printers** command of Windows NT.

Properties

Activate this button to inspect and change the properties of the selected printer.

See also the **File | Print Setup** command for printer changes.

Print to File

Select this check box if you want to store a print file instead of printing on the printer.

The **Print to File** dialog appears. Here, specify the file.

Print Range

Specify the pages you want to print:

Table 4-4. Print Range

Option	Description
All	Prints the entire document.
Pages	- (not selectable)
Selection	Prints the selected element type and its subordinates.

Copies

Edit the number of copies you want to print.

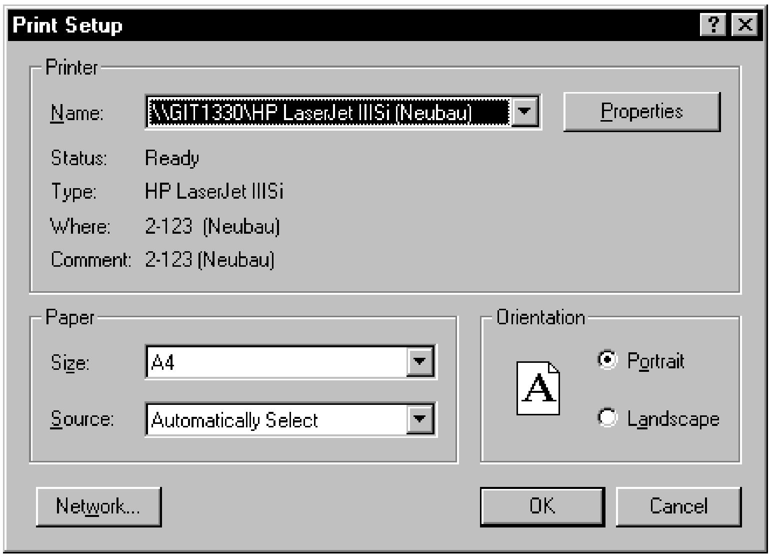
Collate

- (not selectable)

The PROFIBUS Library Editor converts the whole data base or its part as specified into simple list form and prints it.

File | Print Setup …

Use this command to select a printer and a printer connection.



Name: \\GIT1330\HP LaserJet IIISi (Neubau)
Status: Ready
Type: HP LaserJet IIISi
Where: 2-123 (Neubau)
Comment: 2-123 (Neubau)

Size: A4
Source: Automatically Select

● Portrait
○ Landscape

Figure 4-27. Print Setup

Printer

Select one of the current installed printers shown in the drop-down list box you want to use. The static text shows status, type, and location of the selected printer.

You install printers and configure ports using the **Settings | Printers** command of Windows NT.

Properties

Activate this button to inspect and change the properties of the selected printer.

Paper Size and Source

Select the size of paper that the document is to be printed on.

Some printers offer multiple trays for different paper sources. Specify the tray here.

File | 1, 2,...

Use these commands to open one of the recently processed data bases directly, without a dialog.

File | Exit

Use this command to exit the PROFIBUS Library Editor.

If there are unsaved changes, the PROFIBUS Library Editor asks you whether you want to save them before exiting.

Answer **Yes** if you want to save the changes and then leave the PROFIBUS Library Editor.

Answer **No** only if you are sure you want to leave the PROFIBUS Library Editor without saving the changes.

Answer **Cancel** if you changed your mind and you do not want to leave the PROFIBUS Library Editor.

CAUTION

If you answer **No**, you lose all the changes that you made since the last **File | Save**, **File | Save As**, or **File | Exit** command.

4.4.2.2 Edit Menu

Edit | Undo

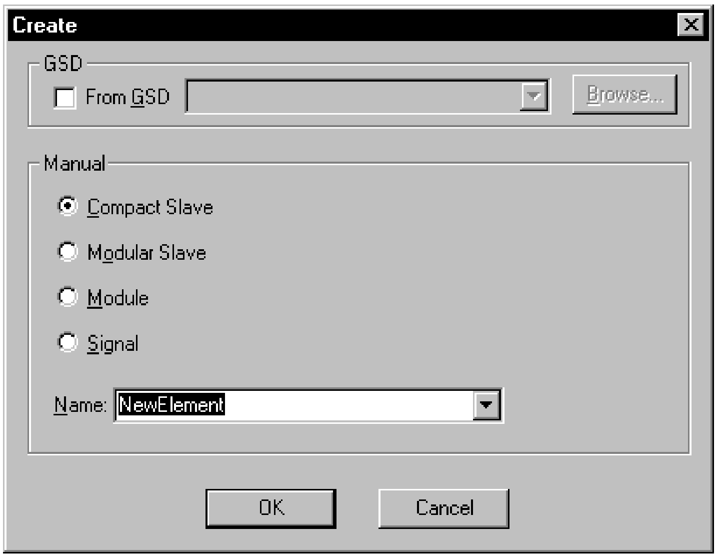
Shortcut: CTRL+Z

Use this command to undo the last editing command or modification in the current data base.

Edit | Create …

Shortcut: INS

Use this command to create new elements within the current data base.



☐ From GSD

● Compact Slave

○ Modular Slave

○ Module

○ Signal

Name: NewElement

Figure 4-28. Create

GSD

Select the **From GSD** option if you want to create elements semi-automatically on the basis of a *.GSD—or for national language versions a *.GS?—file.

In this case, enter the full path and file name in the text box part, or drop down the list box part and select an earlier *.GSD or *.GS? file in the history, or activate the **Browse** button to find and select the *.GSD or *.GS? file.

Unselect the **From GSD** option if you want to create a single element and enter its properties tab by tab by hand.
In this case, select the type of the element to be created in the **Manual** group.

Manual

Use this group if the **From GSD** option is unselected.

Select one of the options to create a corresponding element.
Enter the name of the element to be created in the **Name** text box instead of the default name `NewElement`.
Drop down the list box part to see the earlier used element names.

The PROFIBUS Library Editor creates the new empty element, inserts it in the data base structure in the left pane of the application window and shows its properties in the right pane.

If you have selected the **From GSD** option, the PROFIBUS Library Editor checks the correct format of the selected *.GSD (or *.GS?) file, checks that User_Prm_Data_Len matches to User_Prm_Data on the basis of the *.GSD file. The PROFIBUS Library Editor creates elements on the basis of the *.GSD file, see in Section A.7, Interpretation Decision Tree.

The main steps of automatic DB element creation and the used decision tables are the following:

1. Mapping to DB element types (Section A.1, Mapping of GSD Keywords, Table A-1)

   The PROFIBUS Library Editor creates DB element callnames for slaves, modules, and signals in accordance with Section A.3.1, Allowed Characters for DB Element Callnames and Short Names. The possible violations are solved by character replacement:

   a. Each not allowed character will be substituted by an underline '_'.

   b. If a name is led by a digit, the latter will be substituted by its capital letter counterpart, that is, '0' by 'A', '1', by 'B',… and '9' by 'J'.

2. Interpretation of first byte of configuration data (Section A.6, Interpretation of Configuration and Diagnostic Data, Table A-5 and Table A-6)

3. Interpretation of length byte of configuration data (Section A.6, Interpretation of Configuration and Diagnostic Data, Table A-7)

4. Decision "input or output" (Table A-8 and Table A-10)

5. Generation of data type conversion setting (Table A-9)

6. Interpretation of data types if GSD is based on the PROFIBUS-DPV1 standard (Section A.9, Evaluation of DPV1 Data Type Bytes, Table A-17).

The result is shown in the **Preview** window. See Section 4.4.1.5, Preview Window.

In due case, the PROFIBUS Library Editor creates a *.LST file with syntactic and semantic errors in the C:\PB_LIB directory instead.

Edit | Cut

Shortcut: CTRL+X

Use this command to copy the selected element (slave, module, signal) to the paste buffer and delete it in the data base structure in the left pane of the application window.

Use this command also to copy the selected text from a text field to the paste buffer and delete it in the text field.

Edit | Copy

Shortcut: CTRL+C

Use this command to copy the selected element (slave, module, signal) to the paste buffer and leave it unchanged in the data base structure in the left pane of the application window.

Use this command also to copy the selected text from a text field to the paste buffer and leave it unchanged in the text field.

Edit | Paste

Shortcut: CTRL+V

Use this command to paste the contents of the paste buffer (an element, such as slave, module, signal) at the selected element to the data base structure in the left pane of the application window.

Use this command also to paste the contents of the paste buffer (some text) at the cursor to a text field.

Edit | Delete

Shortcut: DEL

Use this command to delete the element (slave, module, signal) selected in the data base structure in the left pane of the application window.

Use this command also to delete the text selected in a text field.

Edit | Rename

Use this command to bring the focus to the name of the element (slave, module, signal) selected in the data base structure in the left pane of the application window.

Reedit the name.

Close the new name by pressing ENTER or by clicking outside the name field.

4.4.2.3 View Menu

View | Sort Ascending

Use this command to sort items of the **Standard Terminals**, **Process Terminals**, and **Window Names** tabs alphabetically or numerically (as applicable) in ascending order.

View | Sort Descending

Use this command to sort items of the **Standard Terminals**, **Process Terminals**, and **Window Names** tabs alphabetically or numerically (as applicable) in descending order.

View | Next Pane

Shortcut: F6

The commands and keyboard entry refer to the selected pane of the application window.

Initially the left pane is selected.

Use this command to move the selection from one pane to the next (other) one.

NOTE

Pane selection influences also the set of available commands in the menu and in the context menu (to be opened by right mouse click).

View | Split

Use this command to move the selection to the split bar between the panes of the application window, that is, to "seize" the split bar.

Then you can move it with the LEFT and RIGHT key.

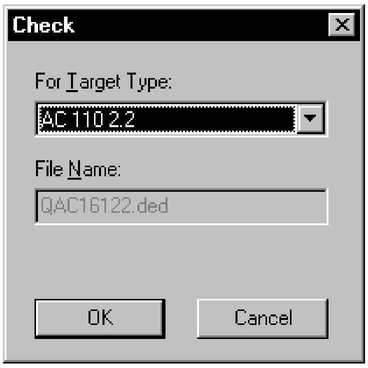
NOTE

You can move the split bar any time by drag and drop with the mouse.

4.4.2.4 Actions Menu

Actions | Check …

Use this command to check your entries at any time during editing and to avoid errors later at compilation of *.DED files.



Check

For Target Type:

AC 110 2.2

File Name:

QAC16122.ded

Figure 4-29. Check

For Target Type

Drop down the list box part to see the available AC 100 Series node types supporting PROFIBUS.

Select the node type, for which you want to check the current data base.

File Name

This control has no significance for the check. (It shows the name of the DB element description source file as fixed for the selected node type.)

The PROFIBUS Library Editor checks the current data base on consistency and shows the result of the check in a message box.

In *AC 100 only* mode it checks also that data type conversion **R** occurs only for new targets like AC 160 2.0, in the *.TED file of which the flag `ProfibusDPRealSupport` is set.

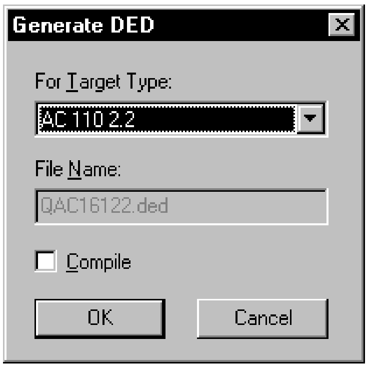
In *AC 400 and AC 100* mode it checks also that `Name` is not longer than 20 (instead of 32) and that no modules and signals occur.

NOTE for Expert Users

Use of this command is mandatory if you import an *.MDB file generated by some other tool than PROFIBUS Library Editor.

Actions | Generate DED for AC 100 …

Activate this command to generate a new DB element description source file for a selected AC 100 Series node type on the basis of the current data base, that is to generate a *.DED file in the ..\DB_LIB\SOURCE directory.



For Target Type: AC 110 2.2

File Name: QAC16122.ded

☐ Compile

Figure 4-30. Generate DED for AC 100

For Target Type

Drop down the list box part to see the available AC 100 Series node types supporting PROFIBUS.

Select the node type, for which you want to generate the new DB element description source file.

File Name

Shows the name of the DB element description source file as fixed for the selected node type.

Compile

Select this option if you want to generate a new DB element description source file and also to compile it to a binary DB element description file, that is to execute automatically also a subsequent **Actions | Compile DED for AC 100** command.

Unselect this option if you want just to generate a new DB element description source file.

The PROFIBUS Library Editor performs implicitly also a check, see the **Actions | Check** command. If the database is correct in the given preferred mode and for the selected node type, the PROFIBUS Library Editor generates and compiles the *.DED file as applicable and shows the result in a message box.

NOTES

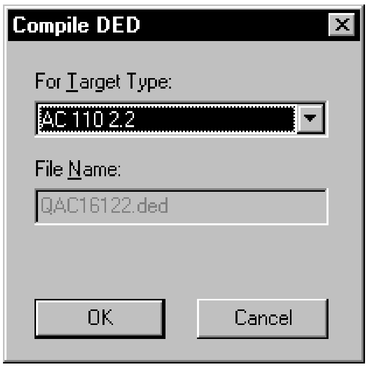
For each node type supporting PROFIBUS, a file name is fixed.
For example, the file name `QAC16122` is fixed for node type AC 110 2.2 supporting PROFIBUS. It is fixed, in fact, by a reference to it in the set of *.TEL files that are part of the AMPL PC and DB Libraries installation and by an item in the [PROFIBUS] group in the FUNCTION.INI file that are part of the Function Chart Builder installation.

The DB element description of slaves and modules will contain their standard terminals—though without default value—even if these terminals do not occur in *.MDB and are not visible by the PROFIBUS Library Editor. See Section A.4, Standard Terminals of Slave Element Types and Section A.5, Standard Terminals of Module Element Types.

Actions | Compile DED for AC 100 …

Activate this command to compile the DB element description source file to a binary DB element description file for a selected AC 100 Series node type and to check that target description files reference it properly.

In other words, to generate a *.DEL file from a *.DED file, the former being in the ..\DB_LIB directory, the latter in the ..\DB_LIB\SOURCE directory.



For Target Type: AC 110 2.2

File Name: QAC16122.ded

Figure 4-31. Compile DED for AC 100

For Target Type

Drop down the list box part to see the available node types supporting PROFIBUS.

Select the node type, for which you want to compile the new DB element description source file.

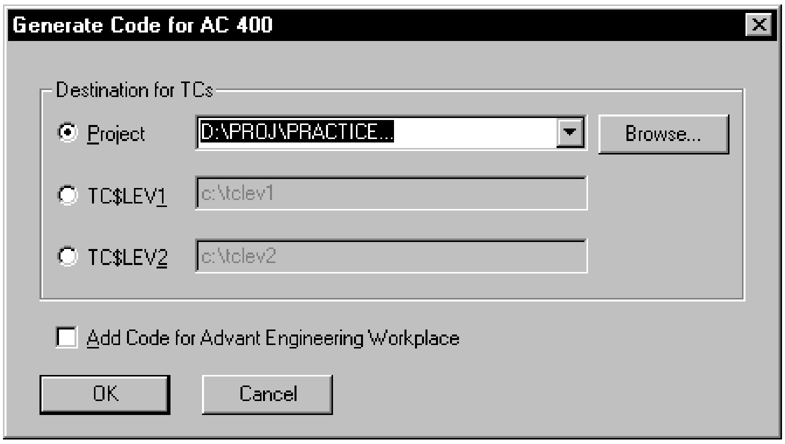
File Name

Shows the name of the DB element description source file as fixed for the selected node type.

The PROFIBUS Library Editor shows the result of *.DED file compilation in a message box.

Actions | Generate Code for AC 400 …

Activate this command to generate type circuit source files and template files from the current data base.



Generate Code for AC 400

Destination for TCs

● Project: D:\PROJ\PRACTICE...

○ TC$LEV1: c:\tclev1

○ TC$LEV2: c:\tclev2

☐ Add Code for Advant Engineering Workplace

Figure 4-32. Generate Code for AC 400

Destination for TCs

Select here the destination for the generated type circuits:

- **Project**
  Set or select the project-specific type circuit directory (..\TCDATA) or browse for it.
- **TC$LEV1** and
- **TC$LEV2**
  The dimmed text fields show the current level1 and level2 type circuit directories. You can change these directories by setting the environment variables TC$LEV1 and TC$LEV2.

Add Code for Advant Engineering Workplace

Select this check box to add some symbols to the P* type circuits. These symbols are necessary if you use the type circuits in conjunction with the Advant Engineering Workplace.

The PROFIBUS Library Editor performs implicitly also a check, see the **Actions | Check** command. If the database is correct and the preferred mode is *AC 400 and AC 100*, the PROFIBUS Library Editor generates the P*.TCS and D*.TCS type circuit source files and the *.AAX and *.BAX template files and shows the result in a message box.

NOTE

The destination of the template file pair is normally the directory `C:\PB_LIB`. You can change it by setting the environment variable PB$LIB.

4.4.2.5 Help Menu

Help | Contents

Use this command to display the opening screen of PROFIBUS Library Editor Help.

The opening screen with **Contents** tab lists the available "how to"-type and reference information on the PROFIBUS Library Editor. The information is hierarchically ordered.

Click on the book symbols 📕 to open them (📖) and on help topic symbols ❓ to jump to final topics of interest.

Click on underlined text to jump to more detailed or related topics.

In the **PROFIBUS Library Editor 2.* - Help** window you can click the **Contents** button whenever you want to return to the contents.

Help | Release Notes

Use this menu command to open the PROFIBUS Library Editor Release Notes containing the following:

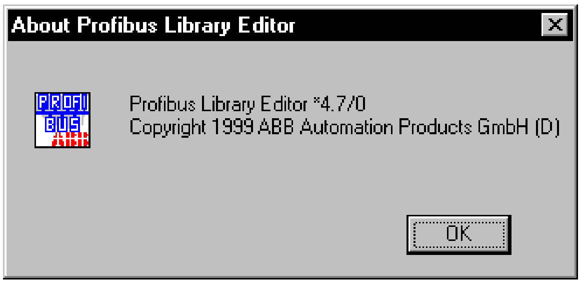
- General information on the PROFIBUS Library Editor version, compatibility, and related documentation
- New functions, improvements, discontinued and replaced functions of the PROFIBUS Library Editor; fixed or known problems, restrictions, and recommendations
- Installation and un-installation of the PROFIBUS Library Editor version; upgrading, backup, and restoration of application data.

NOTE

The Release Notes is a *.PDF file.
To read it, you need Acrobat Reader installed on your PC and the extension PDF associated with the Acrobat Reader.
Acrobat Reader is a free product of Adobe Systems Inc.

Help | About …

Use this command to show the copyright and release notice of PROFIBUS Library Editor.



Profibus Library Editor *4.7/0
Copyright 1999 ABB Automation Products GmbH (D)

Figure 4-33. PROFIBUS Library Editor: About Box

Chapter 5 Maintenance

5.1 Error Messages

The error messages of the PROFIBUS Library Editor appear in dialog boxes containing

- The title **PROFIBUS Library Editor**
- An error message identifier, for example, #12320
- An icon reflecting the severity of the message, for example, a stop sign ✖
- Concise statement of the problem
- A hint to circumvent the problem
- **OK** button to close the dialog box.

Follow the hint.

In case of error messages of other programs than the PROFIBUS Library Editor, consult their documentation.

Appendix A Data Interpretation by PROFIBUS Library Editor

A.1 Mapping of GSD Keywords

On the basis of the *.GSD files one or several DB element types will be created.

See Section A.2, Rules of Creating DB Element Types.

When filling the database semi-automatically, the mapping of GSD keywords to DB element properties is based on the following table:

Table A-1. Mapping of GSD Keywords

GSD Keyword	Description/Remark	Mapping to DB Elements for AC 100	Mapping to PC and DB Elements for AC 400
Modular_Station	If 0, it is a compact station. If 1, it is a modular station.	Type of slave	Type of slave; modular slaves are automatically converted to compact
Vendor_Name	Max. length=32	**Description**	**-**
Model_Name	All characters that are not allowed for DB element callnames in FCB are substituted by '_'. See Section A.3, Allowed Characters for DB Names. Max. length=32.	**Name** of slave element type	**Name** of slave element type. It is mapped to **PBSD:NAME** (max. 20 chars) and **TC name** (max. 1+7chars).
Ident_Number	Identifier	**PBS:IDENTNO**	**PBSD:IDENTNO**
User_Prm_Data_Len	User parameter data length, optional. Default value is 0.	**PBS:UPRMDLEN**	**PBSD:UPRMDLEN**
Max_User_Prm_Data_Len	Max. user parameter data length, optional. Overrides the values of User_Prm_Data_Len.	**PBS:UPRMDLEN**	**PBSD:UPRMDLEN**
User_Prm_Data	User parameter data, optional.	**PBS:UPRMD**	**PBSD:UPRMD**
Ext_User_Prm_Data_Const	Extended user parameter data, constant; optional. Overrides User_Prm_Data.	**PBS:UPRMD**	**PBSD:UPRMD**

Table A-1. Mapping of GSD Keywords

GSD Keyword	Description/Remark	Mapping to DB Elements for AC 100	Mapping to PC and DB Elements for AC 400
String after "Module ="	All characters that are not allowed for DB element callnames in FCB are substituted by '_'. See Section A.3, Allowed Characters for DB Names. Max. length=32.	For a compact station, the string is ignored. For a modular station, it is the name of module element type.	Ignored
Number of bytes after "Module ="	Configuration data length	**PBS:CFGDLEN**	**PBSD:CFGDLEN**
Bytes after "Module ="	Configuration data	**PBS:CFGD**; corresponding number of **PBS:IN*x*** and **PBS:OUT*x*** terminals are created. See Section A.6, Interpretation of Configuration and Diagnostic Data.	**PBSD:CFGD**; corresponding number of **PB-S:VALUE_*x*** and **PB-R:VALUE_*x*** terminals are created. See Section A.6, Interpretation of Configuration and Diagnostic Data.
Ext_Module_Prm_Data_Len (between "Module" and "EndModule")	Module specific user parameter data length, optional. Default value is 0.	*module_name*:**UPRMDLEN**	**PBSD:UPRMDLEN** User parameter data length of the slave is incremented by this value.
Ext_User_Prm_Data_Const (between "Module" and "EndModule")	Module specific user parameter data, optional.	*module_name*:**UPRMD**	**PBSD:UPRMD** Data is appended to the user parameter data of the slave.
Max_Diag_Data_Len	Maximum diagnostic data length, optional	**PBS:MAXDIAGL**; corresponding number of **PBS:DIAG*x*** terminals are created. See Section A.6, Interpretation of Configuration and Diagnostic Data	**PBSD:MAXDIAGL**; corresponding number of **PB-DIAG:VALUE_*x*** terminals are created. See Section A.6, Interpretation of Configuration and Diagnostic Data
Fail_Safe	Optional	**PBS:FAILSAFE**	**PBSD:FAILSAFE**
Min_Slave_Intervall	Minimum slave interval	**PBS:MINSINT**	**PBSD:MINSINT**

Table A-1. Mapping of GSD Keywords

GSD Keyword	Description/Remark	Mapping to DB Elements for AC 100	Mapping to PC and DB Elements for AC 400
-	Watch dog factors	**PBS:WDFACT1** and **2**	**PBS:WDFACT1** and **2**
DPV1_slave	Configuration data in DPV1 format, optional; if found, **DPV1** will be selected in the **Protocol** group.	**PBS:DP_V1** See Section A.6, Interpretation of Configuration and Diagnostic Data	**PBSD:DP_V1** See Section A.6, Interpretation of Configuration and Diagnostic Data

If a keyword is not found in the *.GSD file the default value is used.
See Section A.4, Standard Terminals of Slave Element Types.
There is no message given if default values are used.

A.2 Rules of Creating DB Element Types

Compact Slave

One DB element type `Compact Slave` is created with values read from the GSD file.

Modular Slave

DB element types `Modular Slave`, `Module`, and `Signal` are created.

- Modular Slave

  It includes all values from the GSD file except CFG data.

- Module

  For every keyword "Module" in GSD a Module element type is created.
  The CFG data are added.
  The type of the module is included in the CFG data, see Section A.6, Interpretation of Configuration and Diagnostic Data.

- Signal

  For each module one signal type is created unless the module has own process terminals, that is, unless the process terminals are included in the Process Values *x* part of the module itself.
  Signal type includes process values which are set by interpretation of the CFG data, see Section A.6, Interpretation of Configuration and Diagnostic Data.
  The name of the signal element type is the name of the related module with an appended 'S'.

A.3 Allowed Characters for DB Names

A.3.1 Allowed Characters for DB Element Callnames and Short Names

DB_callname_character ::= *underline* | *letter* | *digit*

underline = `_´

letter = `A´ | `B´ | `C´ | … | `Z´ | `a´ | `b´ | `c´ | … | `z´

digit = `0´ | `1´ | `2´ … | `9´

DB_callname ::= (*underline* | *letter*) { *DB_callname_character* }

DB_shortname ::= *DB_callname_character* { *DB_callname_character* }

NOTE

digit is not allowed at the first position of the DB element callname.
Length of *DB_callname* is 1 to 34 characters.
Length of *DB_shortname* is 1 to 7 characters.

A.3.2 Allowed Characters for DB Terminal Names

DB_terminalname_character ::= *underline* | *letter* | *digit* | *special1*

underline = `_´

letter = `A´ | `B´ | `C´ | … | `Z´ | `a´ | `b´ | `c´ | … | `z´

digit = `0´ | `1´ | `2´ … | `9´

special1 = ` ´ | `/´ | `(´ | `)´ | `=´ | `+´ | `<´ | `>´ | `-´

DB_terminalname ::= (*underline* | *letter*) { *DB_terminalname_character* }

NOTE

digit is not allowed at the first position of the DB element terminal name.
Length of *DB_terminalname* is 1 to 8 characters.

A.3.3 Allowed Characters for Description of DB Element Type

DB_description_character ::= *underline* | *letter* | *digit* | *special1* | *special2*

underline = `_´

letter = `A´ | `B´ | `C´ | … | `Z´ | `a´ | `b´ | `c´ | … | `z´

digit = `0´ | `1´ | `2´ … | `9´

special1 = ` ´ | `/´ | `(´ | `)´ | `=´ | `+´ | `<´ | `>´ | `-´

```
special2        = `!´ | `@´ | `#´ | `$´ | `%´ | `^´ | `&´ | `*´ | `~´ | "`" | `,´ | `.´ | `?´
                  | `;´ | `:´ | `\´ | `|´ | `[´ | `]´ | `{´ | `}´
```

DB_description ::= { *DB_description_character* }

NOTE

digit is not allowed at the first position of description.
Length of *DB_description* is 0 to 20 characters.

A.4 Standard Terminals of Slave Element Types

The following table shows range and default value of the standard terminals of slave element types.
The default value is set for the terminal of a DB element instance if no value is specified.

Table A-2. Standard Terminals of Slave Element Types

Terminal	Range	Default Value
BUS	0..255	0
STATION	0..125	0
DP_V1	0..1	0
IDENTNO	H’0000..H’FFFF	H’0000
MAXDIAGL	6..244	32
FAILSAFE	0..1	0
WDFACT1	0..255	0
WDFACT2	0..255	0
MINSINT	0..32767	0

BUS=0, STATION=0, and POSITION=0 is an undefined address.
(POSITION is fixed to 0 for slaves.)

BUS=0 and STATION=*s* is a valid slave address for all *s*<>0.

BUS=*b* and STATION=0 is a valid slave address for all *b*<>0.

BUS=*b* and STATION=*s* is a valid slave address for all *b*<>0, *s*<>0.

A.5 Standard Terminals of Module Element Types

The following table shows range and default value of the standard terminals of module element types.

The default value is set for the terminal of a DB element instance if no value is specified.

Table A-3. Standard Terminals of Slave Element Types

Terminal	Range	Default Value
BUS	0..255	0
STATION	0..125	0
POSITION	0..32	0

BUS=0, STATION=0, and POSITION=0 is an undefined address.

BUS=0, STATION=0, and POSITION=*p* is an invalid module address for all *p*<>0.

BUS=0 and STATION=*s*, and POSITION=0 is an invalid module address for all *s*<>0.

BUS=*b* and STATION=0, and POSITION=0 is an invalid module address for all *b*<>0.

BUS=*b*, STATION=*s*, and POSITION=0 is an invalid module address for all *b*<>0, *s*<>0.

All other module addresses are valid.

A.6 Interpretation of Configuration and Diagnostic Data

For an overview of the interpretation process, see Section A.7, Interpretation Decision Tree.

Interpretation of Bytes of *.GSD File

To get the module type from the CFG data, the data bytes (octets) are interpreted according to the rules in the following tables.

You can change the module type during creation of the element types on the basis of the *.GSD file.

Numbering of bits of the general GSD byte format:

Table A-4. GSD Byte, Numbering of Bits

7	6	5	4	3	2	1	0

Interpretation of Identifier Byte in *.GSD File

The first byte of the Cfg_data is the identifier byte.

By bits 5 and 4, it defines the format and the way the following bytes are interpreted.

1. Simple format, meaning of bits of the identifier byte:

Table A-5. Simple Format

Bit	Meaning
7	0: consistency over byte or word 1: consistency over whole length
6	0: length in number of bytes 1: length in number of words
5, 4	00: I/O in specific format (see next table) 01: input 10: output 11: output and input
3..0	00: length of (process) data is 1 byte or 1 word … 15: length of (process) data is 16 bytes or 16 words

Or

2. Specific format, meaning of bits of the identifier byte:

Table A-6. Specific Format, Identifier

Bit	Meaning
7, 6	00: free place 01: input, a length byte follows 10: output, a length byte follows 11: output and input, a length byte follows for input and another one for output
5, 4	00: fixed for specific format
3..0	00: no manufacturer-specific data follow 01..14: number of manufacturer-specific (CFG-)data bytes that follow 15: no manufacturer-specific data follow

Interpretation of Length Byte in *.GSD File

If the identifier byte has specific format, the second byte defines the length of the (process) data.

Specific format, meaning of the bits of the length byte:

Table A-7. Specific Format, Length Byte

Bit	Meaning
7	0: consistency over byte or word 1: consistency over whole length
6	0: length in number of bytes 1: length in number of words
5..0	00: length of (process) data is 1 byte or 1 word … 63: length of (process) data is 64 bytes or 64 words

For further interpretation steps, see Section A.7, Interpretation Decision Tree.

A.7 Interpretation Decision Tree

The following figure contains the overview of the decision process of the PROFIBUS Library Editor at the interpretation of manufacturer-specific configuration data (CFG-data) and diagnostic data.

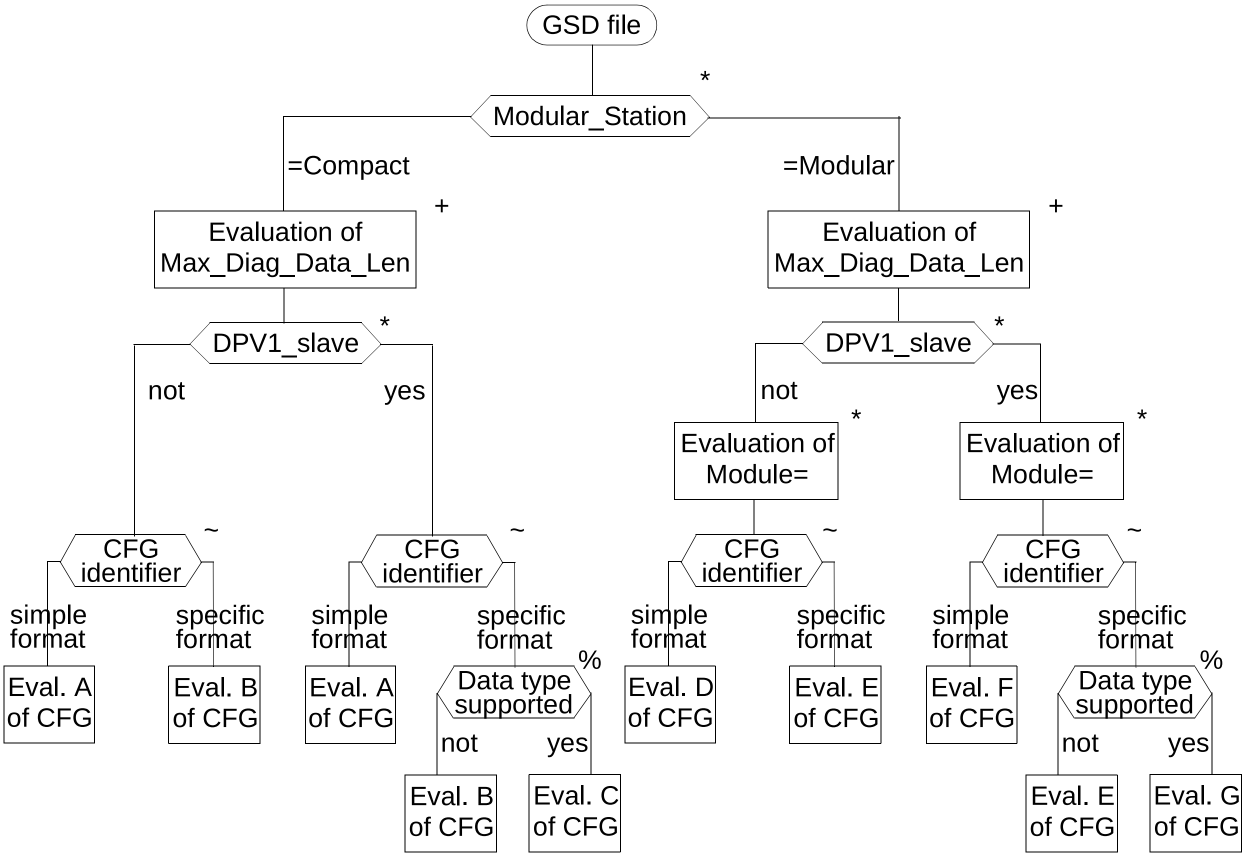


Figure A-1. Interpretation Decision Tree

For information on decisions and actions labelled by a star '*', see Section A.1, Mapping of GSD Keywords.

For information on decisions and actions labelled by a plus '+', see Section A.10, Evaluation of Diagnostic Data.

For information on decisions and actions labelled by a tilde '~', see Section A.6, Interpretation of Configuration and Diagnostic Data.

For information on decisions and actions labelled by a percent '%', see Section A.9, Evaluation of DPV1 Data Type Bytes.

NOTES

The PROFIBUS Library Editor interprets CFG-data and converts data types automatically, that is, it selects the default data type conversion for each case. Especially when `DPV1_slave=0`, it is reasonable that you check the manufacturer specific data and consult the documentation of the slave module.

In due case, add terminals or modify the automatically created terminals. See Section A.8, Possible Conversions of DP Data Types and Section A.9, Evaluation of DPV1 Data Type Bytes when you change the data type conversion by hand.

The actions labelled by "Eval. A" to "Eval. G" are described below:

Evaluation A

Case

Compact Slave station
with configuration data in simple format
and DP or DPV1 format

1. Bits 5 and 4 of identifier byte define if the terminal is input, output, or output and input.

Table A-8. Evaluation of Input/Output in Identifier Byte (Simple Format)

Bits 5 and 4 of Identifier Byte (Simple Format)	Properties of Process Terminal to Be Created				
	Name	Data Type Conversion	Data Group	Bit Number	Offset
0 1	IN*x* *x*=1, 2,...	*(1)	read	*	*
1 0	OUT*x* *x*=1, 2,...	*	write	*	*
1 1	OUT*x* and IN*x* *x*=1, 2,...	*	write read	* *	* *

(1) * means: This evaluation does not change this property, it depends on other conditions.

2. Bits 3 to 0 of identifier byte together with bit 6 define the process data length. No length byte is available.
3. If a single process data length retrieved from CFG data is equal or less than 2 bytes or is equal 1 word, digital process terminal process (B <-> Byte Packed, B<->Word Packed) are assumed, else analog terminals (I, I<->DP 2 Bytes).
4. There is no mixture of analog and digital terminals. If the complete CFG data defines any analog terminals, all process terminals are assumed to be analog.
5. To restrict the complete number of process terminals, analog terminals (I, I<->DP 2 Bytes) are assumed if the complete CFG data defines in sum more than 4 bytes or more than 2 words process data.

6. The PROFIBUS Library Editor creates the corresponding IN*x*, or OUT*x*, or IN*x* and OUT*x* terminals for the Slave DB element type, where *x* = 1, 2,... is the order number of the terminal, with the following default data type conversions:

Table A-9. Generation of Data Type Conversion

Condition	**Analog / Digital Terminals**	**Properties of Process Terminal to Be Created**				
		Name	**Data Type Conversion**	**Data Group**	**Bit Number**	**Offset**
Length is 1 or 2 bytes(1)	Digital	*(2)	B<->Byte Packed	*	0..7	0.. Step=1/Byte
Length is 1 word(1)		*	B<->Word Packed	*	0..15	0.. Step=1/Word
Length is 1 byte(1)	Analog	*	I	*	0	0.. Step=1/Byte
Length is 2 bytes; DP(1)		*	I<->DP 2 Bytes	*	0	0.. Step = 2 / 2Bytes
Length is 1 word(1)		*	I	*	0	0.. Step=1/Word
Length is >2 bytes; DPV1		*	I	*	0	0.. Step=1/Byte
Length is >2 even bytes; DP		*	I<->DP 2 Bytes	*	0	0.. Step = 2 / 2Bytes
Length is >2 odd bytes; DP		*	I	*	0	0.. Step=1/Byte
Length is >1 word		*	I*	*	0	0.. Step=1/Word

(1) To distinguish these cases, see steps 3 to 5 above.
(2) * means: This evaluation does not change this property, it depends on other conditions.

You can change the data type conversions later by hand.

For reasonable data conversions see Section A.8, Possible Conversions of DP Data Types and Section A.9, Evaluation of DPV1 Data Type Bytes.

7. No subordinate Module DB element types are created.
No subordinate Signal DB element types are created.

Evaluation B

Case

Compact Slave station
with configuration data in specific format
and in DP format,
or if data type is not supported, also in DPV1 format

1. Bits 7 and 6 of identifier byte define whether the terminals are input, output, or output and input.

Table A-10. Evaluation of Input/Output in Identifier Byte (Specific Format)

Bits 7 and 6 of Identifier Byte (Specific)	Process Terminal Properties				
	Name	Data Type Conversion	Data Group	Bit Number	Offset
0 1	IN*x* *x*=1, 2,...	*[1]	read	*	*
1 0	OUT*x* *x*=1, 2,...	*	write	*	*
1 1	OUT*x* and IN*x* *x*=1, 2,...	*	write read	* *	* *

(1) * means: This evaluation does not change this property, it depends on other conditions.

A length byte or two length bytes are generally available defining the length of process data.

2. Bits 3 to 0 of identifier byte define the length of manufacturer specific data (within CFG data) after the length byte(s).
3. If a single process data length retrieved from CFG data is equal or less than 2 bytes or is equal 1 word, digital process terminal process (B <-> Byte Packed, B<->Word Packed) are assumed, else analog terminals (I, I<->DP 2 Bytes).
4. There is no mixture of analog and digital terminals. If the complete CFG data defines any analog terminals, all process terminals are assumed to be analog.
5. To restrict the complete number of process terminals, analog terminals (I, I<->DP 2 Bytes) are assumed if the complete CFG data defines in sum more than 4 bytes or more than 2 words process data.
6. The PROFIBUS Library Editor creates the corresponding IN*x*, or OUT*x*, or IN*x* and OUT*x* terminals for the Slave DB element type, where *x* = 1, 2,... is the order number of the terminal, with the default data type conversions according to Table A-9.
7. Check manually the manufacturer specific data.
   Consult the documentation of the slave module.

In due case, add terminals or modify the automatically created terminals manually.
See Section A.8, Possible Conversions of DP Data Types when you change the data type conversion.

8. No subordinate Module DB element types are created.
No subordinate Signal DB element types are created.

Evaluation C

Case

Compact Slave station
with configuration data in specific format
and DPV1 format with supported data types

1. Bits 7 and 6 of identifier byte define whether the terminals are input, output, or output and input. See Table A-10.
A length byte or two length bytes are generally available defining the length of process data.
2. Bits 3 to 0 of identifier byte define the length of CFG data after the length byte(s).
3. The PROFIBUS Library Editor creates IN*x*, or OUT*x*, or IN*x* and OUT*x* terminals for the Slave DB element type, where *x* = 1, 2,... is the order number of the terminal, on the basis of the data types explicitly defined in DPV1 format and also supported by AC 110.
See Section A.9, Evaluation of DPV1 Data Type Bytes.
4. No subordinate Module DB element types are created.
No subordinate Signal DB element types are created.

Evaluation D

Case

Modular Slave station
with configuration data in simple format
and in DP format

1. Bits 5 and 4 of identifier byte define whether the terminal is input, output, or output and input. See Table A-8.
2. Bits 3 to 0 of identifier byte together with bit 6 define the process data length.
No length byte is available.
3. If a single process data length retrieved from CFG data is equal or less than 2 bytes or is equal 1 word, digital process terminal process (B <-> Byte Packed, B<->Word Packed) are assumed, else analog terminals (I, I<->DP 2 Bytes).
4. There is no mixture of analog and digital terminals. If the complete CFG data defines any analog terminals, all process terminals are assumed to be analog.
5. To restrict the complete number of process terminals, analog terminals (I, I<->DP 2 Bytes) are assumed if the complete CFG data defines in sum more than 4 bytes or more than 2 words process data.

   The PROFIBUS Library Editor creates the corresponding Module of type DI, DO or AI, AO.

6. The PROFIBUS Library Editor creates the corresponding IN*x*, or OUT*x*, or IN*x* and OUT*x* Signal element types, with one terminal each, where *x* = 1, 2,... is the order number of the terminal, with the default data type conversions according to Table A-9.

   If CFG data contain more than one byte of CFG data interpreted as input and output signals, do not create Signal element types for them. Instead, add the terminals to the Module element type itself.

Evaluation E

Case

Modular Slave station
with configuration data in specific format
and in DP format,
or if data type is not supported, also in DPV1 format

1. Bits 7 and 6 of identifier byte define whether the terminals are input, output, or output and input. See Table A-10. A length byte or two length bytes are generally available defining the length of process data.
2. Bits 3 to 0 of identifier byte define the length of manufacturer specific data (within CFG data) after the length byte(s).
3. If a single process data length retrieved from CFG data is equal or less than 2 bytes or is equal 1 word, digital process terminal process (B <-> Byte Packed, B<->Word Packed) are assumed, else analog terminals (I, I<->DP 2 Bytes).
4. There is no mixture of analog and digital terminals. If the complete CFG data defines any analog terminals, all process terminals are assumed to be analog.
5. To restrict the complete number of process terminals, analog terminals (I, I<->DP 2 Bytes) are assumed if the complete CFG data defines in sum more than 4 bytes or more than 2 words process data.

   The PROFIBUS Library Editor creates the corresponding Module of type DI, DO or AI, AO.
6. The PROFIBUS Library Editor creates the corresponding IN*x*, or OUT*x*, or IN*x* and OUT*x* Signal DB element types, with one terminal each, where *x* = 1, 2,... is the order number of the terminal, with the default data type conversions according to Table A-9.

   If CFG data contain more than one byte of CFG data interpreted as input and output signals, do not create Signal element types for them. Instead, add the terminals to the Module element type itself.
7. Check manually the manufacturer specific data.
   Consult the documentation of the slave module and its submodules.
   In due case, add terminals or modify the automatically created terminals manually.
   See Section A.8, Possible Conversions of DP Data Types when you change the data type conversion.

Evaluation F

Case

Modular Slave station
with configuration data in simple format
and in DPV1 format

1. Bits 5 and 4 of identifier byte define whether the terminal is input, output, or output and input. See Table A-8.
2. Bits 3 to 0 of identifier byte together with bit 6 define the process data length.
   No length byte is available.
3. If a single process data length retrieved from CFG data is equal or less than 2 bytes or is equal 1 word, digital process terminal process (B <-> Byte Packed, B<->Word Packed) are assumed, else analog terminals (I, I<->DP 2 Bytes).
4. There is no mixture of analog and digital terminals. If the complete CFG data defines any analog terminals, all process terminals are assumed to be analog.
5. To restrict the complete number of process terminals, analog terminals (I, I<->DP 2 Bytes) are assumed if the complete CFG data defines in sum more than 4 bytes or more than 2 words process data.

   The PROFIBUS Library Editor creates the corresponding Module of type DI, DO or AI, AO.
6. The PROFIBUS Library Editor creates the corresponding IN*x*, or OUT*x*, or IN*x* and OUT*x* process terminals for the Module element types, where *x* = 1, 2,... is the order number of the terminal.
   See Section A.9, Evaluation of DPV1 Data Type Bytes.

Evaluation G

Case

Modular Slave station
with configuration data in specific format
and DPV1 format with supported data types

1. Bits 7 and 6 of identifier byte define whether the terminals are input, output, or output and input. See Table A-10.
   A length byte or two length bytes are generally available defining the length of process data.
2. Bits 3 to 0 of identifier byte define the length of CFG data after the length byte(s).
3. The PROFIBUS Library Editor creates the modules of type AX.
4. The PROFIBUS Library Editor creates the corresponding IN*x*, or OUT*x*, or IN*x* and OUT*x* process terminals for the module DB element types, where *x* = 1, 2,... is the order number of the terminal, on the basis of the data types.
   See Section A.9, Evaluation of DPV1 Data Type Bytes.

A.8 Possible Conversions of DP Data Types

The following tables show the data type conversions as set in the **Data Type Conversion** column of the **Process Terminals** tab for DP data types (and lengths) 1 Byte, 1 Word, 2 Bytes, 2 Words, *n* Bytes, and *n* Words as defined in the CFG data.

Table A-11. DP Data Type and Length "1 Byte", Conversions

Selection of Data Type Conversion in Process Terminals Tab	Mapping to AMPL Data Type	Number of Terminals	Conversion for AC 100	Conversion for AC 400
B	B	1	allowed, though no exact fit	allowed, though no exact fit
B<->Byte Packed	**B**	**1..8**	**default**	**default**
B<->Word Packed	B	-	not allowed	not allowed
I	I	1	allowed, though value range is limited and no exact fit	allowed, though value range is limited and no exact fit
I<->DP 2 Bytes	I	-	not allowed	not allowed
IL	IL	1	allowed, though value range limited and no exact fit	allowed, though value range limited and no exact fit
R	R	1	allowed, depending on node type, though value range is limited and no exact fit	not allowed

Table A-12. DP Data Type and Length "1 Word", Conversions

Selection of Data Type Conversion in Process Terminals Tab	**Mapping to AMPL Data Type**	**Number of Terminals**	**Conversion for AC 100**	**Conversion for AC 400**
B	B	1	allowed, though no exact fit	allowed, though no exact fit
B<->Byte Packed	B	-	not allowed	not allowed
B<->Word Packed	**B**	**1..16**	**default**	**default**
I	I	1	allowed	allowed
I<->DP 2 Bytes	I	-	not allowed	not allowed
IL	IL	1	allowed, though value range limited and no exact fit	allowed, though value range limited and no exact fit
R	R	1	allowed, depending on node type, though value range is limited and no exact fit	not allowed

Table A-13. DP Data Type and Length "2 Bytes", Conversions

Selection of Data Type Conversion in Process Terminals Tab	**Mapping to AMPL Data Type**	**Number of Terminals**	**Conversion for AC 100**	**Conversion for AC 400**
B	B	1..2	allowed, though no exact fit	allowed, though no exact fit
B<->Byte Packed	**B**	**1..16**	**default**	**default**
B<->Word Packed	B	-	not allowed	not allowed
I	I	1..2	allowed, though value range is limited and no exact fit	allowed, though value range is limited and no exact fit
I<->DP 2 Bytes	I	1	allowed	not allowed
IL	IL	1..2	allowed, though value range is limited and no exact fit	allowed, though value range is limited and no exact fit
R	R	1..2	allowed, depending on node type, though value range is limited and no exact fit	not allowed

Table A-14. DP Data Type and Length "2 Words", Conversions

Selection of Data Type Conversion in Process Terminals Tab	**Mapping to AMPL Data Type**	**Number of Terminals**	**Conversion for AC 100**	**Conversion for AC 400**
B	B	1..2	allowed, though no exact fit	allowed, though no exact fit
B<->Byte Packed	B	-	not allowed	not allowed
B<->Word Packed	B	1..32	allowed	allowed
I	**I**	**1..2**	**default**	**default**
I<->DP 2 Bytes	I	-	not allowed	not allowed
IL	IL	1..2	allowed, though value range is limited and no exact fit	allowed, though value range is limited and no exact fit
R	R	1..2	allowed, depending on node type, though value range is limited and no exact fit	not allowed

Table A-15. DP Data Type and Length "n Bytes", Conversions

Selection of Data Type Conversion in Process Terminals Tab	**Mapping to AMPL Data Type**	**Number of Terminals**	**Conversion for AC 100**	**Conversion for AC 400**
B	B	1..*n*	allowed, though no exact fit	allowed, though no exact fit
B<->Byte Packed	B	1..8**n*	allowed	allowed
B<->Word Packed	B	-	not allowed	not allowed
I	**I**	**1..*n***	**default if *n* is odd**, though value range is limited and no exact fit	**default**, though value range is limited and no exact fit
I<->DP 2 Bytes	**I**	**1..*n*/2**	**default if *n* is even**	not allowed
IL	IL	1..*n*	allowed, though value range is limited and no exact fit	allowed, though value range is limited and no exact fit
R	R	1..*n*	allowed, depending on node type, though value range is limited and no exact fit	not allowed

Table A-16. DP Data Type and Length "n Words", Conversions

Selection of Data Type Conversion in Process Terminals Tab	Mapping to AMPL Data Type	Number of Terminals	Conversion for AC 100	Conversion for AC 400
B	B	1..*n*	allowed, though no exact fit	allowed, though no exact fit
B<->Byte Packed	B	-	not allowed	not allowed
B<->Word Packed	B	1..16*n*	allowed	allowed
I	**I**	**1..*n***	**default**	**default**
I<->DP 2 Bytes	I	-	not allowed	not allowed
IL	IL	1..*n*	allowed, though value range is limited and no exact fit	allowed, though value range is limited and no exact fit
R	R	1..*n*	allowed, depending on node type, though value range is limited and no exact fit	not allowed

NOTES to Table A-11 to Table A-16

All data type conversions are offered in the **Data Type Conversion** column of the **Process Terminals** tab for every terminal even if they are not reasonable and in the tables "not allowed".

The values in the Number of Terminals column of the tables are limits. The actual number of terminals (of the DB element and P* type circuit to be generated) depends on the way the PROFIBUS-DP device fills the telegrams with useful data. For example, in case of a device serving 12 digital input channels and sending the process values packed in a word, you select **B<->Word packed** and configure 12 DI terminals with bit numbers 0 to 11, though a word could carry maximum 16 Boolean values.

n is the length, that is, number of bytes or words, as defined in CFG data.

default shows the data type conversion selected by the PROFIBUS Library Editor at automatic element creation on the basis of a PROFIBUS-DP *.GSD file.

Value ranges of related PROFIBUS-DP and AMPL data types are different.
The differences result in limiting the value range of the "larger" type to that of the "smaller" one. For example, when converting a Byte to and from an I(nteger), the value range of the latter is (has to be) limited to -128..127.
Conversion may imply also "wasting" bits or bytes. For example, at the conversion of a Byte to and from one B(oolean) data, 7 bits of the former remain unused, or at the conversion of an integer carried by a Word to and from an I(nteger)L(ong), two bytes of the latter remain unused.

A.9 Evaluation of DPV1 Data Type Bytes

If the identifier byte has specific format and `DPV1_slave` keyword was found, the manufacturer-specific (CFG-)data bytes will be interpreted, they define the data types of process data.

In the following table, we show also the corresponding AMPL data types.

Specific format, meaning of the data type bytes after the length byte:

Table A-17. Data Type Byte

Value /Code	PROFIBUS-DPV1 Data Type	Number of Bytes	Mapping to/ AMPL Data Type	Number of Terminals
1	Boolean	1	B	1
2	Integer8	1	I	1
3	Integer16	2	I	1
4	Integer32	4	IL	1
5	Unsigned8	1	I	1
6	Unsigned16	2	IL	1
7	Unsigned32	4	IL	1
8	Floating Point	4	R	1
9	Visible String	1, 2, 3,…	I	1, 2, 3,…
10	Octet String	1, 2, 3,…	I	1, 2, 3,…
11	Date	7	not yet supported	-
12	Time Of Day	4	not yet supported	-
13	Time Difference	4	not yet supported	-
14	Time Of Day	6	not yet supported	-
15	Time Difference	6	not yet supported	-
16..31	(reserved)			
32..63	user specific			
64..255	(reserved)			

NOTE

If `DPV1_slave` keyword was found and a code of the upper table is detected which is not yet supported, then the PROFIBUS Library Editor handles the DB elements as PROFIBUS DP element and not as PROFIBUS DPV1 element. This means that it selects Protocol DP, sets standard terminal DB_V1 to 0, interprets CFG data for DP.

A.10 Evaluation of Diagnostic Data

The number of diagnostic data—and diagnostic terminals—is specified by the keyword `Max_Diag_Data_Len` in the *.GSD file.

The minimum number diagnostic terminals is 6 even if the GSD includes `Max_Diag_Data_Len` < 6.

In case of AC 100 Series, the PROFIBUS Library Editor creates one DIAG*x* terminal in the Slave DB element type for each byte of diagnostic data.

In case of AC 400 Series, the PROFIBUS Library Editor creates one VALUE_*x* terminal in the PB-DIAG PC element for each byte of diagnostic data. You can limit their appearance in P* type circuits.

Table A-18. Evaluation of Diagnostic Data

Properties of Diagnostic Terminal to Be Created				
Name	**Data Type Conversion**	**Data Group**	**Bit Number**	**Offset**
DIAG*x* *x*=1, 2,...	I	diag	0	0..243 Step=1/byte

A.11 Signal Types Depending on Module Type

The following table shows the dependence between module type and signal types and the item designations of their instances:

Table A-19. Signal Types Depending on Module Type

Module Type	**Item Designation of Module Instance**	**Signal Type**	**Item Designation of Signal Instances**
AI	AIPB*x*	AI	AIPB*x.y*
AO	AOPB*x*	AO	AOPB*x.y*
DI	DIPB*x*	DI	DIPB*x.y*
DO	DOPB*x*	DO	DOPB*x.y*
AX	AXPB*x*	AI, AO, DI, DO	AXPB*x.y*
DX	DXPB*x*	AI, AO, DI, DO	DXPB*x.y*

A.12 Background of Data Type Conversion in AC 100 Series

General

Data have an AMPL data type, such as B, I, IL, and R, when processed in AC 100 (and AC 400) Series systems. This data type shows the way you process a given signal in your application: as B(oolean) or I(nteger), and so on.

Clearly, if you have a digital I/O device in your Advant system, you work with B(oolean) signals in your application, if you have an analog I/O device, you work with I(nteger), I(nteger)L(ong), or R(eal) signals, and so on.

On the other hand, data in the PROFIBUS-DP devices can have data types Byte and Word, and data in the PROFIBUS-DPV1 devices can have data types Boolean, Integer8, Integer16, Unsigned8, and so on.

NOTE

When programming an application, especially when working with extended DB elements, you handle signals of Advant analog I/O devices as R(eal).

PROFIBUS-DP analog I/O devices work with data type Word carrying in that case analog information coded as integer. In contrast to the PROFIBUS-DP interface of older AC 100 Series node types such as AC 110 2.1, the PROFIBUS-DP interface of newer ones such as AC 160 2.0 does convert the integer to and from R(eal).

Some of the PROFIBUS-DPV1 analog I/O devices work with data type Floating Point that corresponds to R(eal). In these cases, you can profit from comfortable programming techniques with any node type and it is easy to adapt existing applications to configurations with PROFIBUS I/O.

PROFIBUS-DP or DPV1 data type is fixed and described in the *.GSD file and in the documentation of each slave device and is reflected in the CFG Data.

The PROFIBUS-DP interface of the processor module, such as PM634, handles data from and to the slaves on the PROFIBUS also on the basis of their CFG Data.

Data types and conversion cases are summarized for the PROFIBUS-DP interface of processor module PM634 in Table A-20 and Table A-21.

Data Transfer and Conversion in PROFIBUS-DP Interface of PM634

In the PROFIBUS-DP interface of PM634, **data from and to the PROFIBUS** are buffered data by data in a 4-byte buffer. In the following table, the column Data in PROFIBUS-DP Interface shows how many bytes are filled with useful data starting from the lowest byte of the buffer. The rest of the buffer, that is, the currently unused higher bytes, are cleared.

Table A-20. Data on PROFIBUS-DP Interface of PM634

Handled by PROFIBUS Slaves as		Data in PROFIBUS-DP Interface
PROFIBUS-DP Data Types	**PROFIBUS-DPV1 Data Types**	
Byte	Boolean, Integer8, Unsigned8, Visible String(*i*), Octet String(*i*)(1)	1 byte
Word	Integer16, Unsigned16	2 bytes
-	Integer32, Unsigned32, Floating Point	4 bytes

(1) … String (i) means the i-th byte of a Visible String or Octet String.

The **data from and to the application** are transferred to and from the buffer in a natural one-to-one way or they can be converted in the PROFIBUS-DP interface of processor module PM634.

Table A-21. Data to and from the Application

Data in PROFIBUS-DP Interface(1)	Conversion	Handled in Application as AMPL Data Type	Comment
1 byte	none	B, I, IL	Limited value range for I and IL
	B<->Byte Packed	B (8 times)	In DPV1 only for Octet String(1)
2 separate bytes(2)	I<->DP 2 Bytes	I	Not for DPV1 slave
2 bytes	none	I	Not recommended for Unsigned16
	none	B, IL	Limited value range for IL
	B<->Word Packed	B (16 times)	Not for DPV1 slave
4 bytes	none	IL	Check value range for Unsigned32
	none	R	

(1) Note that this column contains the same items as the last column of the previous table.

(2) These two bytes stem from two consecutive transfers to or from the 4 byte buffer in the PROFIBUS-DP interface of processor module PM634

INDEX

Symbols
# (hash mark) 5-1
*.AA 4-17
*.AAX 4-15
*.BA 4-17
*.BAX 4-15
*.DED 4-14
*.GSD 3-2, 4-8, 4-11, 4-13, A-1, A-6, A-9, A-21
*.LST 4-15
*.MDB 4-10
*.PLE 4-14
*.TCS 4-15
*.TEL 4-15

A
abbreviations 1-5
AC 100 only (mode) 4-1
AC 100 Series usage 3-1
AC 400 and AC 100 (mode) 4-1
AC 400 Series usage 3-3
Access 97 1-7, 4-10
Actions | Check 4-58
Actions | Compile DED for AC 100 4-60
Actions | Generate Code for AC 400 4-61
Actions | Generate DED for AC 100 4-59
Actions Menu 4-58
Advant Controller 100 Series 1-7, 3-5 to 3-6, 4-2, 4-6, 4-14 to 4-15
Advant Controller 400 Series 1-7, 3-6 to 3-7, 4-3, 4-7, 4-15 to 4-16
Allowed Characters for DB Names A-4
AMPL Control Configuration 1-6
Application Builder 4-10

B
Background of Data Type Conversion in AC 100 Series A-22
BUS A-5 to A-6

C
callname A-4
capacity 3-6
CFG tab (of compact slave) 4-30
CFG-data A-9
checklist 3-6 to 3-7
CI541 4-7
commands 3-1, 3-3
Compact Slave 4-1, 4-21
compatibility 2-1
compiling the PROFIBUS library 4-14
configuration in AMPL Control Configuration 4-4
considerations 3-1
context menu
  of left pane 4-40
  of right pane 4-41
controls
  of compact slave 4-21
  of modular slave 4-34
  of module 4-36
  of PROFIBUS Library Editor 4-18
  of signal 4-39
conventions 1-3
conversion of data type 4-24, A-16, A-22
copyright 4-62

D
D* type circuit 4-9, 4-17
DAO 3.5 4-10
data flow 3-1, 3-3
Data Type A-16, A-20, A-22
database configuration 4-5
DB element 4-4 to 4-5, A-3
  hierarchy 4-5, 4-7
  library 3-5
Default Value A-5 to A-6
Description 4-20, A-4
diagnostic data A-9, A-21
documentation 1-4

E
Edit | Copy 4-56
Edit | Create 4-54
Edit | Cut 4-56
Edit | Delete 4-56
Edit | Paste 4-56
Edit | Rename 4-56
Edit | Undo 4-54
Edit Menu 4-54
editing
  GSD of PROFIBUS devices and variants 4-11
  the PROFIBUS library 4-13
element library 4-4, 4-10

Element Type 4-19
error messages 5-1
Evaluation of Diagnostic Data A-21
Evaluation of DPV1 Data Type Bytes A-20
executable 1-7
exiting the PROFIBUS Library Editor 4-15

F
features 1-7
File | 1, 2, ... 4-53
File | Compact Database 4-50
File | Exit 4-53
File | Export 4-50
File | Import 4-47
File | New 4-44
File | Open 4-45
File | Print 4-51
File | Print Setup 4-53
File | Save 4-48
File | Save As 4-48
File | Skip Modifications 4-49
File Menu 4-44
files 4-10
Function Chart Builder 4-4, 4-12

G
generating
  template 4-15
  type circuits 4-15

H
Help | About 4-62
Help | Contents 4-62
Help | Release Notes 4-62
Help Menu 4-62
hierarchy
  of DB elements 4-5, 4-7
  of type circuits 4-8

I
Identifier byte A-6
installation 2-1, 4-10
Interpretation Decision Tree A-9
Interpretation of Configuration and Diagnostic Data A-6

L
left pane of application window 4-18
Length byte A-7

M
manufacturer-specific configuration data A-9
Mapping of GSD Keywords A-1
master 3-5
model data base 1-7
Modular Slave 4-1, 4-34
Module 4-36
module pool 4-18
Module Type 4-1, A-21
Modules tab (of modular slave) 4-35

N
Name 4-19

O
overview 1-6, 4-4

P
P* type circuit 4-9, 4-17
pane 4-18 to 4-19, 4-40 to 4-41
PB 4-5
PB$LIB 4-11
PB_LIB 4-11
PB-DIAG 4-7
PB-R 4-7
PB-S 4-7
PBS 4-7
PBSD 4-7
performance 3-6
PM5## 4-7
PM634 4-5
Possible Conversions of DP Data Types A-16
preferred mode 4-1
Preview window 4-41
Process Terminals tab
  of compact slave 4-23
  of module 4-37
  of signal 4-39
product
  name 2-1
  overview 1-6
  verification 2-1
  version 2-1
PROFIBUS library
  compiling 4-14
  editing 4-13
  working with 4-15

PROFIBUS Library Editor
    controls of 4-18
    exiting 4-15
PROFIBUS type circuits 4-7
PROFIBUS.MDB 4-10
PROFIBUS-DP 1-6, A-16
PROFIBUS-DPV1 1-6, A-20
programming in AMPL Control Configuration 4-4
project-specific type circuit directory 4-15
Protocol 4-20

Q
QAC16122.DCL 4-15
QAC16122.DED 4-15
QAC16122.DEL 4-15

R
release
    directory 4-10
    notice 4-62
releases 1-5
Representations of PROFIBUS Configurations 3-5
requirements 1-1
right pane of application window 4-19
Rules of Creating DB Element Types A-3

S
saving previous work 4-12
Short Name 4-1, 4-19, A-4
signal 4-39
signal pool 4-19
Signal Types Depending on Module Type A-21
Signals tab (of module) 4-38
slave 3-5
slave variant 4-3, 4-12
Standard Terminals of Module Element Types A-5
Standard Terminals of Slave Element Types A-5
Standard Terminals tab
    of compact slave 4-21
    of modular slave 4-34
    of module 4-36
starting the PROFIBUS Library Editor 4-13
start-up 3-6
STATION A-5 to A-6
structure tree of slaves 4-18
system overview 3-1, 3-3

T
TC$LEV1 and TC$LEV2 4-11, 4-15
TCLEV1 and TCLEV2 4-11
TCs generated by PLE, working with 4-16
template 4-15
    generating 4-15
terminal name A-4
terms 1-5
type circuit 4-5
    generating 4-15
    hierarchy 4-8
    library 3-5

U
UPRM tab (of compact slave) 4-28
usage
    for Advant Controller 100 Series 4-2
    for Advant Controller 400 Series 4-3
user interface 1-8

V
View | Next Pane 4-57
View | Sort Ascending 4-57
View | Sort Descending 4-57
View | Split 4-57
View Menu 4-57

W
Window Names tab (of compact slave) 4-32
Windows NT 1-6, 2-1
working
    with TCs generated by PLE 4-16
    with the PROFIBUS library 4-15

3BDS 100 585R1

November 1999